Refine Search

Search Results -

Terms	Documents
L1 and e-commerce and rules	4

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L2

Refine Search

Recall Text

Clear

Interrupt

Search History

DATE: Thursday, April 01, 2004 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=OR			
L2	L1 and e-commerce and rules	4	L2
L1	706/47.ccls.	323	L1

END OF SEARCH HISTORY

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 4 of 4 returned.

☐ 1. Document ID: US 20030023573 A1

Using default format because multiple data bases are involved.

L2: Entry 1 of 4 File: PGPB Jan 30, 2003

PGPUB-DOCUMENT-NUMBER: 20030023573
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030023573 A1

TITLE: Conflict-handling assimilator service for exchange of rules with merging

PUBLICATION-DATE: January 30, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Chan, Hoi Yeung	Stamford	CT	US	
Grosof, Benjamin N.	Newton	MA	US	

US-CL-CURRENT: 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 2. Document ID: US 20020198856 A1

L2: Entry 2 of 4 File: PGPB Dec 26, 2002

PGPUB-DOCUMENT-NUMBER: 20020198856
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020198856 A1

TITLE: Minimization of business rules violations

PUBLICATION-DATE: December 26, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Feldman, Jacob	Edison	NJ	US	
Korolov, Alexander	Cliffside Park	NJ	US	
Meshcheryakov, Semen	Edison	NJ	US	
Shor, Stanislav	Edison	NJ	US	

US-CL-CURRENT: 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 3. Document ID: US 20020065745 A1

L2: Entry 3 of 4 File: PGPB May 30, 2002

PGPUB-DOCUMENT-NUMBER: 20020065745
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020065745 A1

TITLE: Rule-based personalization framework for integrating recommendation systems

PUBLICATION-DATE: May 30, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Rainsberger, Joseph B.	Toronto		CA	
Tin, Ramiah Kwok-Fai	Pickering		CA	
Tong, Tack	Scarborough		CA	

US-CL-CURRENT: 705/27; 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 4. Document ID: US 6070149 A

L2: Entry 4 of 4 File: USPT May 30, 2000

US-PAT-NO: 6070149
DOCUMENT-IDENTIFIER: US 6070149 A

TITLE: Virtual sales personnel

DATE-ISSUED: May 30, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Tavor; Onn	Ramat Hasharon			IL
Avraham; Gila Ben	Netania			IL
Shevchenko; Vadim	Netania			IL

US-CL-CURRENT: 705/26; 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
L1 and e-commerce and rules	4

Display Format: - Change Format

Previous Page Next Page Go to Doc#

Refine Search

Search Results -

Terms	Documents
L3 and empirical and engine	21

Database:
US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Search: L4

Refine Search | Recall Text | Clear | Interrupt

Search History

DATE: Thursday, April 01, 2004 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=OR			
L4	L3 and empirical and engine	21	L4
L3	e-commerce and rules and recommendations	395	L3
L2	L1 and e-commerce and rules	4	L2
L1	706/47.ccls.	323	L1

END OF SEARCH HISTORY

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 21 of 21 returned.

☐ 1. Document ID: US 20040054572 A1

Using default format because multiple data bases are involved.

L4: Entry 1 of 21 File: PGPB Mar 18, 2004

PGPUB-DOCUMENT-NUMBER: 20040054572
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20040054572 A1

TITLE: Collaborative filtering

PUBLICATION-DATE: March 18, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Oldale, Alison	London		GB	
Oldale, John	London		GB	
Reenen, John Van	London		GB	
Campbell, Michael	London		GB	

US-CL-CURRENT: 705/10

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 2. Document ID: US 20040034652 A1

L4: Entry 2 of 21 File: PGPB Feb 19, 2004

PGPUB-DOCUMENT-NUMBER: 20040034652
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20040034652 A1

TITLE: System and method for personalized search, information filtering, and for generating recommendations utilizing statistical latent class models

PUBLICATION-DATE: February 19, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Hofmann, Thomas	Barrington	RI	US	
Puzicha, Jan Christian	Albany	CA	US	

US-CL-CURRENT: 707/102

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 3. Document ID: US 20030036944 A1

L4: Entry 3 of 21 File: PGPB Feb 20, 2003

PGPUB-DOCUMENT-NUMBER: 20030036944
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030036944 A1

TITLE: Extensible business method with advertisement research as an example

PUBLICATION-DATE: February 20, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Lesandrini, Jay William	Greenfield	IN	US	
Smith, Margaret Paige	Indianapolis	IN	US	
White, Jane	Fishers	IN	US	
Santiago, John Anthony	Racine	WI	US	
Schachter, Andrew	Toronto		CA	

US-CL-CURRENT: 705/10; 705/14

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 4. Document ID: US 20020107853 A1

L4: Entry 4 of 21 File: PGPB Aug 8, 2002

PGPUB-DOCUMENT-NUMBER: 20020107853
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020107853 A1

TITLE: System and method for personalized search, information filtering, and for generating recommendations utilizing statistical latent class models

PUBLICATION-DATE: August 8, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Hofmann, Thomas	Barrington	RI	US	
Puzicha, Jan Christian	Albany	CA	US	

US-CL-CURRENT: 707/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 5. Document ID: US 20020099596 A1

L4: Entry 5 of 21 File: PGPB Jul 25, 2002

PGPUB-DOCUMENT-NUMBER: 20020099596
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020099596 A1

TITLE: Dynamic ratemaking for insurance

PUBLICATION-DATE: July 25, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Geraghty, Michael Kevin	Marietta	GA	US	

US-CL-CURRENT: 705/10

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 6. Document ID: US 20020091991 A1

L4: Entry 6 of 21 File: PGPB Jul 11, 2002

PGPUB-DOCUMENT-NUMBER: 20020091991
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020091991 A1

TITLE: Unified real-time microprocessor computer

PUBLICATION-DATE: July 11, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Castro, Juan Carlos	Miami	FL	US	

US-CL-CURRENT: 717/106

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 7. Document ID: US 20020065745 A1

L4: Entry 7 of 21 File: PGPB May 30, 2002

PGPUB-DOCUMENT-NUMBER: 20020065745
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020065745 A1

TITLE: Rule-based personalization framework for integrating recommendation systems

PUBLICATION-DATE: May 30, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Rainsberger, Joseph B.	Toronto		CA	
Tin, Ramiah Kwok-Fai	Pickering		CA	
Tong, Tack	Scarborough		CA	

US-CL-CURRENT: 705/27; 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 8. Document ID: US 20020042733 A1

L4: Entry 8 of 21 File: PGPB Apr 11, 2002

PGPUB-DOCUMENT-NUMBER: 20020042733
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020042733 A1

TITLE: Enhancements to business research over internet

PUBLICATION-DATE: April 11, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Lesandrini, Jay William	Greenfield	IN	US	
Smith, Margaret Paige	Indianapolis	IN	US	
White, Jane	Fishers	IN	US	

US-CL-CURRENT: 705/10; 705/14

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 9. Document ID: US 20020010679 A1

L4: Entry 9 of 21 File: PGPB Jan 24, 2002

PGPUB-DOCUMENT-NUMBER: 20020010679
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020010679 A1

TITLE: Information record infrastructure, system and method

PUBLICATION-DATE: January 24, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Felsher, David Paul	Trumbull	CT	US	

US-CL-CURRENT: 705/51; 705/3

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 10. Document ID: US 20010052108 A1

L4: Entry 10 of 21 File: PGPB Dec 13, 2001

PGPUB-DOCUMENT-NUMBER: 20010052108
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20010052108 A1

TITLE: SYSTEM, METHOD AND ARTICLE OF MANUFACTURING FOR A DEVELOPMENT ARCHITECTURE FRAMEWORK

PUBLICATION-DATE: December 13, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
BOWMAN-AMUAH, MICHEL K.	COLORADAO SPRINGS	CO	US	

US-CL-CURRENT: 717/100

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 11. Document ID: US 6687696 B2

L4: Entry 11 of 21 File: USPT Feb 3, 2004

US-PAT-NO: 6687696
DOCUMENT-IDENTIFIER: US 6687696 B2

TITLE: System and method for personalized search, information filtering, and for generating recommendations utilizing statistical latent class models

DATE-ISSUED: February 3, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hofmann; Thomas	Barrington	RI		
Puzicha; Jan Christian	Albany	CA		

US-CL-CURRENT: 707/6; 707/4

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 12. Document ID: US 6662357 B1

L4: Entry 12 of 21 File: USPT Dec 9, 2003

US-PAT-NO: 6662357
DOCUMENT-IDENTIFIER: US 6662357 B1

TITLE: Managing information in an integrated development architecture framework

DATE-ISSUED: December 9, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 717/120

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 13. Document ID: US 6629081 B1

L4: Entry 13 of 21 File: USPT Sep 30, 2003

US-PAT-NO: 6629081
DOCUMENT-IDENTIFIER: US 6629081 B1
**** See image for Certificate of Correction ****

TITLE: Account settlement and financing in an e-commerce environment

DATE-ISSUED: September 30, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Cornelius; Richard D.	Santa Monica	CA		
Stepniczka; Andreas	San Francisco	CA		
Chu; Kevin	Atlanta	GA		

US-CL-CURRENT: 705/30

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 14. Document ID: US 6615166 B1

L4: Entry 14 of 21 File: USPT Sep 2, 2003

US-PAT-NO: 6615166
DOCUMENT-IDENTIFIER: US 6615166 B1

TITLE: Prioritizing components of a network framework required for implementation of technology

DATE-ISSUED: September 2, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Guheen; Michael F.	Tiburon	CA		
Mitchell; James D.	Manhattan Beach	CA		
Barrese; James J.	San Jose	CA		

US-CL-CURRENT: 703/27; 703/26, 709/220, 709/223, 709/231, 717/140, 719/316

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 15. Document ID: US 6536037 B1

L4: Entry 15 of 21 File: USPT Mar 18, 2003

US-PAT-NO: 6536037
DOCUMENT-IDENTIFIER: US 6536037 B1
**** See image for Certificate of Correction ****

TITLE: Identification of redundancies and omissions among components of a web based architecture

DATE-ISSUED: March 18, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Guheen; Michael F	Tiburon	CA		
Mitchell; James D.	Manhattan Beach	CA		
Barrese; James J.	San Jose	CA		

US-CL-CURRENT: 717/151; 703/2, 709/231

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 16. Document ID: US 6519571 B1

L4: Entry 16 of 21 File: USPT Feb 11, 2003

US-PAT-NO: 6519571
DOCUMENT-IDENTIFIER: US 6519571 B1

TITLE: Dynamic customer profile management

DATE-ISSUED: February 11, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Guheen; Michael F.	Tiburon	CA		
Mitchell; James D.	Manhatten Beach	CA		
Barrese; James J.	San Jose	CA		

US-CL-CURRENT: 705/14

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw De

☐ 17. Document ID: US 6473794 B1

L4: Entry 17 of 21 File: USPT Oct 29, 2002

US-PAT-NO: 6473794
DOCUMENT-IDENTIFIER: US 6473794 B1

TITLE: System for establishing plan to test components of web based framework by displaying pictorial representation and conveying indicia coded components of existing network framework

DATE-ISSUED: October 29, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Guheen; Michael F.	Tiburon	CA		
Mitchell; James D.	Manhattan Beach	CA		
Barrese; James J.	San Jose	CA		

US-CL-CURRENT: 709/223; 709/224

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 18. Document ID: US 6405364 B1

L4: Entry 18 of 21 File: USPT Jun 11, 2002

US-PAT-NO: 6405364
DOCUMENT-IDENTIFIER: US 6405364 B1

TITLE: Building techniques in a development architecture framework

DATE-ISSUED: June 11, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 717/101; 717/102, 717/120, 717/124

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

☐ 19. Document ID: US 6370573 B1

L4: Entry 19 of 21 File: USPT Apr 9, 2002

US-PAT-NO: 6370573
DOCUMENT-IDENTIFIER: US 6370573 B1

TITLE: System, method and article of manufacture for managing an environment of a development architecture framework

DATE-ISSUED: April 9, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 709/223

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 20. Document ID: US 6324647 B1

L4: Entry 20 of 21 File: USPT Nov 27, 2001

US-PAT-NO: 6324647
DOCUMENT-IDENTIFIER: US 6324647 B1
** See image for Certificate of Correction **

TITLE: System, method and article of manufacture for security management in a development architecture framework

DATE-ISSUED: November 27, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO	80918	

US-CL-CURRENT: 713/201; 709/223, 713/153

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 21. Document ID: US 6256773 B1

L4: Entry 21 of 21 File: USPT Jul 3, 2001

US-PAT-NO: 6256773
DOCUMENT-IDENTIFIER: US 6256773 B1

TITLE: System, method and article of manufacture for configuration management in a development architecture framework

DATE-ISSUED: July 3, 2001

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY

Bowman-Amuah; Michel K. Colorado Springs CO

US-CL-CURRENT: 717/121; 707/203, 717/168

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
L3 and empirical and engine	21

Display Format: - Change Format

Previous Page Next Page Go to Doc#

Refine Search

Search Results -

Terms	Documents
rule adj system and empirical and recommendation adj system	1

Database: US Pre-Grant Publication Full-Text Database; US Patents Full-Text Database; US OCR Full-Text Database; EPO Abstracts Database; JPO Abstracts Database; Derwent World Patents Index; IBM Technical Disclosure Bulletins

Search: L5

Refine Search | Recall Text | Clear | Interrupt

Search History

DATE: Thursday, April 01, 2004 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=OR			
L5	rule adj system and empirical and recommendation adj system	1	L5
L4	L3 and empirical and engine	21	L4
L3	e-commerce and rules and recommendations	395	L3
L2	L1 and e-commerce and rules	4	L2
L1	706/47.ccls.	323	L1

END OF SEARCH HISTORY

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 1 of 1 returned.

☐ 1. Document ID: US 20020065745 A1

Using default format because multiple data bases are involved.

L5: Entry 1 of 1 File: PGPB May 30, 2002

PGPUB-DOCUMENT-NUMBER: 20020065745
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020065745 A1

TITLE: Rule-based personalization framework for integrating recommendation systems

PUBLICATION-DATE: May 30, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Rainsberger, Joseph B.	Toronto		CA	
Tin, Ramiah Kwok-Fai	Pickering		CA	
Tong, Tack	Scarborough		CA	

US-CL-CURRENT: 705/27; 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
rule adj system and empirical and recommendation adj system	1

Display Format: - Change Format

Previous Page Next Page Go to Doc#

Google Web Images Groups News Froogle[New!] more »
+"e-commerce" +"rule system" +"recomme Search Advanced Search Preferences

Web Results **1** - **10** of about **25** for **+"e-commerce" +"rule system" +"recommendation"**. (**0.65** seconds)

Artificial Intelligence and Soft Computing 2000
... Performance of Collaborative **Recommendation** by using ... Component-based Hierarchical **Rule System** for Business ... D. Elliott: Deploying **E-Commerce** Enabling Technology ...
www.informatik.uni-trier.de/~ley/ db/conf/asc/asc2000.html - 26k - Cached - Similar pages

AI for EC
... In this paper, we describe a **rule system** and an interactive ... They serve many types of **E-commerce** applications, from direct product **recommendation** for an ...
www.cs.umbc.edu/aiec/contents.shtml - 38k - Cached - Similar pages

[PDF] Evolving Disclosure, E-Commerce Reshaping Municipal Industry
File Format: PDF/Adobe Acrobat - View as HTML
... MUNICIPAL BOND DIVISION ROUNDTABLE Evolving Disclosure, **E-Commerce** Reshaping Municipal ... 23, 2001 SEC approved final **rule SYSTEM** SPECIFICATION PUBLICATION DATE ...
www.bondmarkets.com/newsletters/2001/bmn201.pdf - Similar pages

[PDF] Recommender systems and Personalization Techniques
File Format: PDF/Adobe Acrobat - View as HTML
... The majority of **e-commerce** websites currently use ... House.com is using an association **rule system**. ... item-based collaborative filtering **recommendation** algorithms [4 ...
www.macalester.edu/~gveletsianos/ Recommender_Systems.pdf - Similar pages

[PDF] Active rules for XML: A new paradigm for E-services
File Format: PDF/Adobe Acrobat
... that active rules can be effective for the im- plementation of **e-commerce** services. ... required for the design and implementation of an active **rule system** for XML ...
portal.acm.org/ ft_gateway.cfm?id=767136&type=pdf&coll=portal&dl=ACM&CFID=11111111&CFT... - Similar pages

2001
... The production **rule system** and the authoring tool ... Keywords: Personalization, **Recommendation** System Case-Based ... on issues related to **e-commerce** applications in ...
tcc.itc.it/publications/2001/ - 60k - Cached - Similar pages

[PDF] Agent-driven Online Business in Virtual Communities
File Format: PDF/Adobe Acrobat - View as HTML
... sophisticated strategies based on a **rule system** the agents ... two scenarios for B2C **e- commerce** are outlined ... Through word-of-mouth **recommendation** from another ...
www.casba-market.org/CASBA_HICSS33mas.pdf - Similar pages

Pushing Reactive Services to XML Repositories using Active Rules

Sponsored Links

E-Commerce System
Streamline the sale of complex products.
www.technicon.com

E-commerce System
Java Community Process: the Center of Java Technology Developments !
www.sun.com

Shopping Cart Software
Powerful, Reliable, Easy-to-use. We deliver results. Free 30-day trial.
www.Fortune3.com/e-commerce/

E-Commerce Research
Read Electronic **Commerce** research from the leading IT analyst firms.
www.analystviews.com

Instant Activation
No Setup Fee - 98% approved shopping cart available, installed
www.rcktech.net

See your message here...

... more or less concerned with **e-commerce** applications ... it is evident that the **rule system** hereby proposed ... Version 1.0 W3C Proposed **Recommendation**", September 2000 ...
www10.org/cdrom/papers/623/ - 65k - Cached - Similar pages

RuleML Homepage
... inference rules were marked up for **E-Commerce** and were ... to a rule in some "logic of **recommendation**". ... Since RuleML should help **rule-system** interoperation, (XSLT ...
thais.cs.ece.ntua.gr/NLP/CGs/ruleml.html - 74k - Cached - Similar pages

[PDF] The Power of Personalization: Customer Collaboration and Virtual ...
File Format: PDF/Adobe Acrobat - View as HTML
... implemented by a pre-defined **rule system** which combines ... **E-Commerce** systems track and store compound profiles which ... MyYahoo and the Amazon **Recommendation** Center ...
e-business.fhbb.ch/.../4950a19acb63938bc1256c110063856a/ $FILE/schubert_koch.pdf - Similar pages

Gooogle ▶
Result Page: 1 2 3 **Next**

+"e-commerce" +"rule system" +" Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

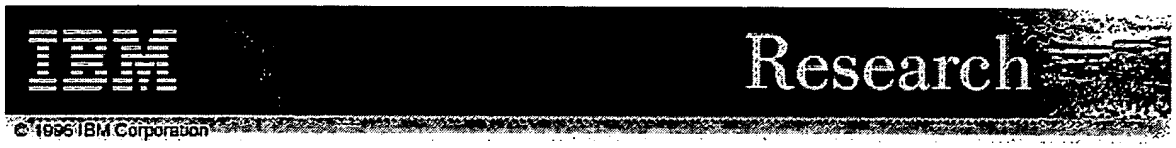
Google Home - Advertising Solutions - Business Solutions - About Google

©2004 Google

This is G o o g l e's cache of http://www.research.ibm.com/rules/commonrules-overview.html.
G o o g l e's cache is the snapshot that we took of the page as we crawled the web.
The page may have changed since that time. Click here for the current page without highlighting.

Google is not affiliated with the authors of this page nor responsible for its content.

These search terms have been highlighted: **rule system e commerce**



Overview of IBM CommonRules 1.0 Alpha Release

Up to Business Rules project home page

IBM releases CommonRules 1.0: business rules for the Web.

July 30, 1999. IBM announces the release of CommonRules 1.0 today, a Java library that brings **e-commerce** business rules up to Internet speed. CommonRules enables Web communication of executable business rules between enterprises using heterogeneous rule systems, and enables incremental specification of executable business rules by non-programmer business domain experts.

CommonRules 1.0 is available free for download at AlphaWorks (http://alphaworks.ibm.com). It was developed by IBM Research's Business Rules for **E-Commerce** project team (http://www.research.ibm.com/rules/). CommonRules is 100 percent pure Java, and runs on all Java platforms including Windows 95, NT, and 98.

CommonRules provides innovative XML inter-operability and prioritized conflict handling capabilities. These modularly augment a wide variety of rule-based systems and programming mechanisms already available in the market. The CommonRules 1.0 release includes API's for developers to enhance Java or non-Java applications. It also includes extensive documentation and example rule sets.

Using CommonRules, a seller website/application can communicate its business policy rules about pricing, promotions, customer service provisions for refunds and cancellation, ordering lead time, and other contractual terms & conditions, to a customer application/agent, even when the seller's rules are implemented using a different **rule system** (e.g., OPS5-style production rules) than the buyer's rules are implemented in (e.g., Prolog). The customer application/agent can then understand and assimilate those rules into its own business logic, and automatically execute those rules to make plans or decisions. Similarly, using CommonRules, a customer can communicate to its suppliers the customer's Requests For Quotation/Proposals, including detailed conditions and policies expressed in rules, e.g., in supply chain settings.

CommonRules also helps enable non-programmer business-domain experts, for example marketing managers, to easily modify the executable business rules incrementally at run-time. For example, on Wednesday a customer retention manager specifies a personalized pricing rule that gives loyal customers a ten percent price discount, and on Thursday a cash flow manager specifies that late-to-pay customers get no discount. These two rules conflict for customer Joe, who is loyal but late-to-pay. CommonRules detects the conflict automatically and raises an alarm. A sales manager resolves the conflict by simply adding at run-time a further rule specifying that the first rule has priority over the second. The new rule set will then automatically cover future customers like Joe, without need for any additional specification.

CommonRules is being piloted in EECOMS, a $29 Million 3-year industry consortium development effort on inter-enterprise supply chain integration for manufacturing. CommonRules is also being demonstrated internationally at several scientific conferences' peer-reviewed technical programs. The research website (http://www.research.ibm.com/rules/; contact project leader Benjamin Grosof grosof@us.ibm.com or Hoi Chan hychan@us.ibm.com) includes extensive discussion of **e-commerce** application areas for CommonRules' capabilities, including for contracts, negotiation, auctions, personalization and promotions, and security authorization. The research website includes published papers and detailed research reports, as well as overviews and talk slides.

More background about business rules

A major trend happening in object-oriented application development is the separation of business logic from data access and application logic: the same business logic may be used in multiple applications, and should be changeable rather than being buried and intertwined with data and application specific functions.

Rules as a way to specify business logic has the advantage of combining automatic executability with a relatively high level of human understandability, i.e., a high conceptual level and a "declarative" (rather than only procedural) semantics. The latter enables non-programmers, especially business-domain experts such as marketing managers, to specify business rules, and to modify them relatively easily and often.

CommonRules' role is to complement and enhance the functionality of the various rule-based systems and programming mechanisms already available in the market.

Technical overview of CommonRules

CommonRules provides a common "interlingua" rule representation for exchange of rules between heterogeneous rule representations employed in various rule-based applications. CommonRules defines and supports a new XML rule interchange format for rules, called Business Rules Markup Language (BRML), that corresponds to this interlingua. Rules may be exchanged as XML, directly as Java objects, or in other string formats; the rule-based applications need not be in Java. The interlingua enables two or more applications/websites/agents, that use heterogeneous rule systems/languages, to exchange rules in a fully declarative fashion, while preserving deep semantics so that received rules can be fully assimilated, i.e., understood and executed with the same semantics intended by the sender. The interlingua has a semantics based on Logic Programs (in the sense of declarative knowledge representation, not just Prolog). This semantics captures a common core shared by most commercially important rule systems, including SQL / relational database systems, Prolog and logic programming systems, production rule systems (OPS5 heritage), and event-condition-action rule systems. In particular, this semantics abstracts away from whether rule inferencing/execution is performed in a

forward (data-driven) versus backward (query-driven) direction. CommonRules includes sample translators between the BRML XML interchange format and several existing rule systems. Developers (e.g., **rule system** vendors) can write their own such translators, relatively easily. Future versions of CommonRules plan to include more such sample translators.

CommonRules enhances this core representation with a new form of prioritized conflict handling for rules, called Courteous Logic Programs, that enables rules to be specified and maintained in a far more modular fashion, more similar to the "common sense" style in which rules are expressed in natural language. Courteous Logic Programs enforce consistency with respect to specified mutual-exclusion integrity constraints, using specified partially-ordered priorities between conflicting rules. Rule sets can then be modified far more often by simply adding rules, without the need to modify previous rules. CommonRules provides a practical breakthrough combination of expressive power, declarativeness, computational efficiency/tractability, and software modularity in prioritized conflict handling. It includes a Courteous Compiler that implements the Courteous enhancement via a pre-processor that can be added modularly to a variety of existing commercial rule systems. This Courteous Compiler transforms a Courteous Logic Program into a semantically equivalent but expressively simpler ordinary Logic Program (lacking priorities or mutual-exclusion integrity constraints) of the kind widely implemented in today's commercial rule engines/systems. CommonRules includes a sample forward-reasoning inference engine that uses the courteous compiler as a pre-processor, and thus supports the Courteous expressive enhancement. Developers (e.g., **rule system** vendors) can similarly use the Courteous Compiler to write their own enhanced rule engines.

Further details about CommonRules

- BRML complements and extends ANSI-draft Knowledge Interchange Format (KIF), and provides the first XML encoding for KIF. In version 1.0, only a broad sub-case of KIF is represented: clauses. Future versions of BRML plan to represent more of KIF. BRML goes beyond KIF to support logical non-monotonicity, including the negation-as-failure, the most practically important form of negation, and prioritized conflict handling. CommonRules includes a sample translator to/fro KIF's existing (non-XML) string format.

- A near-future version of CommonRules plans to also include declaratively clean procedural attachments, in the patented manner pioneered in IBM's earlier Agent Building Environment (from the same research team).

- The deep semantics of (declarative) logic programs specifies what set of conclusions is sanctioned/entailed by any given set of premise rules (and facts).

- Inferencing in Courteous Logic Programs is guaranteed to be computationally tractable (worst-case polynomial-time), given the Datalog constraint and a bounded number of logical variables per rule. The Courteous Compiler has worst-case cubic computational complexity (time and space). Previous approaches to prioritized rules with consistency enforced for mutual exclusions were computationally intractable (exponential time, i.e., NP-hard or worse) to perform rule inferencing/execution.

- CommonRules was demonstrated on July 20-21, 1999, as part of the refereed technical program at AAAI-99, the National Conference on Artificial Intelligence, held in Orlando, Florida. It will also be demonstrated on July 31, 1999, as part of the refereed technical program at AMEC-99, the Agent-Mediated **E-Commerce** workshop at IJCAI-99, the International Joint Conference on Artificial Intelligence, held in Stockholm, Sweden.

For further information, please contact: Benjamin Grosof at grosof@us.ibm.com

Last update: **8-09-99**

[IBM Research home page]

[IBM home page | Order | Search | Contact IBM | Help | (C) | (TM)]

This is **G o o g l e**'s cache of http://xml.coverpages.org/ruleML.html.
G o o g l e's cache is the snapshot that we took of the page as we crawled the web.
The page may have changed since that time. Click here for the current page without highlighting.

Google is not affiliated with the authors of this page nor responsible for its content.

These search terms have been highlighted: **java rule system**



Last modified: November 30, 2002

Technology Reports

Rule Markup Language (RuleML)

[November 17, 2000] The RuleML Initiative represents a collaborative research effort by an international team of participants seeking to develop shared Rule Markup Language (RuleML). The project is consciously related to other standards work, including Mathematical Markup Language (MathML), DARPA Agent Markup Language (DAML), Predictive Model Markup Language (PMML), Attribute Grammars in XML (AG-markup), and Extensible Stylesheet Language Transformations (XSLT). From the web site description: "The participants of the RuleML Initiative constitute an open network of individuals and groups from both industry and academia. We are not commencing from zero but have done some work related to rule markup or have actually proposed some specific tag set for rules. Our main goal is to provide a basis for an integrated rule-markup approach that will be beneficial to all involved and to the rule community at large. This shall be achieved by having all participants collaborate in establishing translations between existing tag sets and in converging on a shared rule-markup vocabulary. This RuleML kernel language can serve as a specification for immediate rule interchange and can be gradually extended - possibly together with related initiatives - towards a proposal that could be submitted to the W3C. Rules can be stated (1) in natural language, (2) in some formal notation, or (3) in a combination of both. Being in the third, 'semiformal' category, the RuleML Initiative is working towards an XML-based markup language that permits Web-based

HOSTED BY
OASIS
SPONSORED BY
Global eXchange Services
INNODATA ISOGEN
SAP
Sun microsystems
COVER PAGES NEWSLETTER
Receive weekly news updates from Managing Editor, Robin Cover.
Subscribe >>
View Archives >>

rule storage, interchange, retrieval, and firing/application. Rules in (and for) the Web have become a mainstream topic since inference rules were marked up for E-Commerce and were identified as a Design Issue of the Semantic Web, and since transformation rules were put to practice for document generation from a central XML repository (as used here). Rules have also continued to play an important role in Intelligent Agents and AI shells for knowledge-based systems, which need a Web interchange format, too. The Rule Markup Initiative has taken initial steps towards defining a shared Rule Markup Language (RuleML), permitting both forward (bottom-up) and backward (top-down) rules in XML for deduction, rewriting, and further inferential-transformational tasks. The initiative started during PRICAI 2000, as described in the Original RuleML Slide, and was launched in the Internet on 2000-11-10. A complementary effort coordinates the development of **Java** rule engines. A Rule Markup Workshop is planned in conjunction with the third International Conference on Electronic Commmerce, ICEC2001, in Vienna, Austria, in October 2001."

"RuleML largely grows out of the design approach and design criteria of Business Rules Markup Language (BRML) which was developed in my previous work at IBM Research and which is implemented in IBM CommonRules 4 available under free trial license from IBM alphaWorks. The design approach and design criteria of CommonRules and BRML are described in [Grosof et al., 1999; Grosof and Labrou, 2000], and in the documentation in the CommonRules download package. BRML's expressive class is situated courteous logic programs, *i.e.*, declarative logic programs with negation-as-failure, (limited) classical negation, prioritized conflict handling, and disciplined procedural attachments for queries and actions. RuleML differs in several significant respects from its BRML predecessor, however. One respect is that it defines a family of DTDs. More deeply, however, these differences largely revolve around 'Webizing' the KR: (1) URIs for logical vocabulary and knowledge subsets; (2) labels for rules/rulebases, import/export; (3) headers: metadata describes the XML document's expressive class; (4) procedural attachments using Web protocols/services, queries or actions via CGI/servlets/SOAP/..." [from the IJCAI 2001 Workshop paper of B. Grosof]

"Engines: "One or more rule engines will be needed for executing RuleML modules. On 2000-11-15, the RuleML Initiative thus joined forces with the **Java** Specification Request JSR-000094 **Java** Rule

Engine API. This cooperation will enable a direct cross-fertilization between the complementary specifications of the open XML-based Rule Markup Language and of the **Java** runtime API for rule engines."

References:

- RuleML web site

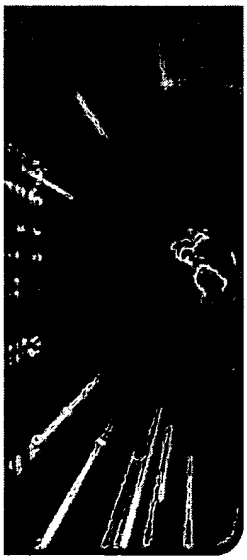
- [November 05, 2002] CommonRules version 3.3 from IBM alphaWorks now offers "improved performance; faster processing speed due to a better fact-matching algorithm; a new object-mapping system; improved documentation; and additional built-in functions. CommonRules is a rule-based framework for developing rule-based applications with major emphasis on maximum separation of business logic and data, conflict handling, and interoperability of rules. It is a pure **Java** library, and it provides a platform that enables the rapid development of rule-based applications through its situated rule engine via dynamic and real-time connection with business objects. CommonRules can be integrated with existing applications at a specific point of interest, or it can be used to create applications composed only of rules. CommonRules uses a sematically-rich rule language called CLP (Courteous Logic Program) to enable direct conflict resolution through conditional mutual exclusion and prioritized override. It contains a set of APIs for efficient application integration, as well as data and function bindings. Also included is a prototype for rule interlingua, which is currently based on CLP; later, it will be based on RuleML (the proposed standard rule format) in order to enable interoperability of different rules... CommonRules provides innovative XML interoperability and prioritized conflict-handling capabilities. These modularly augment a wide variety of rule-based systems and programming mechanisms already available in the market. CommonRules 3.3 includes an API set for enhancing **Java** or non-**Java** applications. It also includes extensive documentation and example rule sets. Using CommonRules, a seller Web site or application can communicate in XML its business policy rules about pricing, promotions, customer service provisions for refunds and cancellation, ordering lead time, and other contractual terms and conditions, to a

customer application or agent, even when the seller's rules are implemented using a different **rule system** (such as OPS5-style production rules) than that in which the buyer's rules are implemented (such as Prolog). The customer application or agent can then understand and assimilate those rules into its own business logic, and it can automatically execute those rules to make plans or decisions."

- [September 21, 2001] ICEC 2001 Workshop on Semantic Web-based E-Commerce and Rules Markup Language. October 31 - November 4th, 2001. Vienna, Austria. "... an exchange of information and ideas, and to facilitate the discussion of current and emerging topics related to E-Commerce, the Semantic Web and Rules Markup Languages."

- [October 22, 2001] "The Rule Markup Language: RDF-XML Data Model, XML Schema Hierarchy, and XSL Transformations." By Harold Boley. Invited presentation at the 14th International Conference of Applications of Prolog (INAP2001), October 20-22, 2001. The University of Tokyo, Sanjo Conference Hall, Japan. 16 pages. See also the slides.

- [May 19, 2001] "RuleML DTDs." By Harold Boley, Benjamin Grosof, and Said Tabet. Version 0.7 (2001-01-25) or later. "This is a preliminary DTD draft for RuleML. Each DTD in the evolving hierarchy corresponds to a specific RuleML sublanguage. The DTDs use a modularization approach similar to the one in XHTML in order to offer appropriate flexibility and accomodate different implementations and approaches. We will write a technical report on this system of RuleML DTDs..."



- RuleML for W3C face-to-face Technical Plenary Meeting.. By Benjamin Grosof. February 26 to March 2, 2001. Materials include (1) The One-pager: A Flyer announcing the Rules Birds Of a Feather session held at the W3C meeting, and summarizing what RuleML is. (2) 15-minute Overview Talk -- Slides: RuleML Overview Slides presented at the Birds Of a Feather session held at the W3C meeting. (3) More Technical Details on the Strawman-version RuleML Syntax -- Talk Slides: RuleML

Syntax: Examples and DTDs -- Talk Slides presented at the Birds Of a Feather session held at the W3C meeting."

- "Design Rationale of RuleML: A Markup Language for Semantic Web Rules." By Harold Boley, Said Tabet, and Gerd Wagner. "This paper lays out the design rationale of RuleML, a rule markup language for the Semantic Web. We give an overview of the RuleML Initiative as a Web ontology effort. Subsequently, the modular syntax and semantics of RuleML and the current RuleML 0.8 DTDs are presented (focusing on the Datalog and URI sublanguages). Then we discuss negation handling, priorities/evidences, as well as agents and RuleML. We next proceed to RuleML implementations via XSLT and rule engines. In our conclusions, we continue to explore the bigger picture of ontologies and discuss some requirements for a future RuleML. An appendix shows our Semantic Web scenario in the insurance industry... To accomodate the various (Web) rule-user communities from Knowledge-Based Systems to Intelligent Agents to E-Commerce, a modular hierarchy of sublanguages will be discussed. Rule extensions will concern first-class URIs, Web-suited negations, labelings, certainties/priorities, and packages. The Initiative also examines where current description methods and implementation techniques (e.g., XML DTDs vs. Schemas and C- vs. **Java**-based rule engines) are sufficient for such rule markup and where they would need revisions/extensions. This paper further attempts to contribute to some open issues of Notation 3 (N3) and DAML-Rules in relation to RuleML. Finally, by studying issues of combining rules and taxonomies via sorted logics, description logics, or frame systems, the paper also touches on the US-European proposal DAML+OIL..."

- [March 26, 2002] "Standardizing XML Rules: Rules for E-Business on the Semantic Web." Invited Presentation (45-minutes, presentation, with slides in PDF format). By Benjamin N. Grosof (MIT Sloan Professor in E-Commerce Information Technology). August 5, 2001. Presented at the Workshop on E-business and the Intelligent Web at the International Joint Conference on Artificial Intelligence (IJCAI-01). See also the short paper; preliminary prose outline of the talk,

and appears in the Workshop Proceedings. The principal topic of discussion is the Rule Markup Language (RuleML). [alt URL for paper; cache]

- [May 19, 2001] "Standardizing XML Rules." By Benjamin N. Grosof (MIT Sloan School of Management, Cambridge, MA, USA. Email: bgrosof@mit.edu or grosof@cs.stanford.edu). Invited paper for the IJCAI 2001 Workshop on E-Business and the Intelligent Web [August 5 2001], part of the Seventeenth International Joint Conference on Artificial Intelligence. ['The author provides an overview of current efforts to standardize rules knowledge representation in XML, with special focus on the design approach and criteria of RuleML, an emerging standard. With Harold Boley of DFKI (Germany) and Said Tabet of Nisus Inc. (USA), Benjamin N. Grosof leads an early-phase standards effort on a markup language for exchange of rules in XML, called RuleML (Rule Markup Language); the goal of this effort is eventual adoption as a Web standard, *e.g.*, via the World Wide Web Consortium'] "RuleML is, at its heart, an XML syntax for rule knowledge representation (KR), that is inter-operable among major commercial rule systems. It is especially oriented towards four commercially important families of rule systems: SQL (relational database), Prolog, production rules (cf. OPS5, CLIPS, Jess) and Event-Condition-Action rules (ECA). These kinds of rules today are especially found embedded in Object-Oriented (OO) systems, and are often used for business process connectors / workflow. These four families of rule systems all have common core abstraction: declarative logic programs (LP). 'Declarative' here means in the sense of KR theory. Note that this supports both backward inferencing and forward inferencing. RuleML is actually a family (lattice) of rule KR expressive classes: each with a DTD (syntax) and an associated KR semantics (KRsem). These expressive classes form a generalization hierarchy (lattice). The KRsem specifies what set of conclusions are sanctioned for any given set of premises. Being able to define an XML syntax is relatively straightforward. Crucial is the semantics (KRsem) and the choice of expressive features. The motivation to have syntax for several different expressive classes, rather than for one most general expressive class, is that: precision facilitates and maximizes effective interoperability,

given heterogeneity of the rule systems/applications that are exchanging rules. The kernel representation in RuleML is: Horn declarative logic programs. Extensions to this representation are defined for several additional expressive features: (1) negation: negation-as-failure and classical negation; (2) prioritized conflict handling: *e.g.*, cf. courteous logic programs; (3) disciplined procedural attachments for queries and actions: *e.g.*, cf. situated logic programs; (4) equivalences, equations, and rewriting; (5) and other features as well. In addition, RuleML defines some useful expressive restrictions (*e.g.*, Datalog, facts-only, binary-relations-only), not only expressive generalizations... In January 2001, we released a first public version of a family of DTDs for several flavors of rules in RuleML. This was presented at the W3C's Technical Plenary Meeting held February 26 to March 2, 2001. Especially since then, RuleML has attracted a considerable degree of interest in the R&D community. Meanwhile, the design has been evolving to further versions." [cache]

- Rule Markup Initiative. By Harold Boley. Presented at The Sixth Pacific Rim International Conference on Artificial Intelligence (PRICAI 2000) Melbourne, Australia, on 31 August 2000.

- RuleML Announcement. From Harold Boley and Said Tabet. 10-November-2000.

- JSR-000094 **Java** Rule Engine API. "This specification defines a **Java** runtime api for rule engines; it targets the J2EE and J2SE platforms...This specification assumes the existence of a parallel effort to specify an open, XML-based rule language. The result of this effort will be an XML Schema that will be registered at a site such as xml.org. [...] The API prescribes an object model and a set of fundamental rule engine operations. The object model and set of operations are based upon the assumption that most clients will need to be able to execute a basic multi-step rule engine cycle, which consists of parsing rules, adding objects to an engine, firing rules and getting resultant objects from the engine. The object model and the set of operations also support variations of the basic cycle, particularly variations that would occur in J2EE server deployments. A primary input to a rule engine is a collection

of rules called a ruleset. The rules in a ruleset are expressed in a rule language. This specification defines api support for parsing rulesets that have been authored in vendor-specific rule languages. Additionally, to help simplify the task of creating rule-authoring tools and because business-to-business exchange of rules is anticipated, this specification defines api support for parsing rulesets that have been authored in XML-based, vendor-independent rule languages."

- "Tutorial on Knowledge Markup Techniques." By Harold Boley, Stefan Decker, and Michael Sintek. '07-Sep-2000'. Also available as PDF and Postscript.

- Contacts: Benjamin N. Grosof (MIT Sloan School of Management), Harold Boley. and Said Tabet

- See also:
  - "STARLab ORM Markup Language (ORM-ML)"
  - "DARPA Agent Mark Up Language (DAML)"
  - "Agent-Oriented Rule Markup Language (AORML)"
  - "Extensible Rule Markup Language (XRML)"
  - "Simple Rule Markup Language (SRML)
  - Business Rules Markup Language (BRML)
  - XML Belief Network File Format (Bayesian Networks)
  - Predictive Model Markup Language (PMML)
  - Relational-Functional Markup Language (RFML)
  - Ontology and Conceptual Knowledge Markup Languages
  - Simple HTML Ontology Extensions (SHOE)
  - XOL - XML-Based Ontology Exchange Language
  - Description Logics Markup Language (DLML)
  - Case Based Markup Language (CBML)
  - Artificial Intelligence Markup Language (AIML)
  - XML and Attribute Grammars

Document URL: http://xml.coverpages.org/ruleML.html

All alphaWorks Searc

IBM home | Products & services | Support & downloads | My account

Select a country

developerWorks

alphaWorks
Emerging Technologies

Application Development
Autonomic Computing
Collaboration
Eclipse Technology
Grid Computing
Java Technology
Privacy & Security
Web Services
Wireless
XML

- Select -

find a technology

Licensing
Discussion Forums
alphaWorks in the News
alphaWorks Newsletters

Newsletters
Subscribe to the aW flash newsletter:
your e-mail Go
Text HTML
More newsletters

Free Download>> VOICE TOOLKIT

Develop DB2 UDB apps on Windows -Tools & resources

alphaWorks emerging technologies 1 April
Contact aW About

alphaWorks > App Dev > CommonRules > Overview

CommonRules

Date Posted: July 30, 1999

Download now
License now

Update: October 30, 2002
Version 3.3: Improved performance; faster processing speed due to a better fact-matching algorithm; a new object-mapping system; improved documentation; and additional built-in functions.

What is CommonRules?

CommonRules 3.3 is a rule-based framework for developing rule-based applications with major emphasis on maximum separation of business logic and data, conflict handling, and interoperability of rules. It is a pure Java™ library, and it provides a platform that enables the rapid development of rule-based applications through its situated rule engine via dynamic and real-time connection with business objects. CommonRules can be integrated with existing applications at a specific point of interest, or it can be used to create applications composed only of rules.

CommonRules uses a sematically-rich rule language called CLP (Courteous Logic Program) to enable direct conflict resolution through conditional mutual exclusion and prioritized override. It contains a set of APIs for efficient application integration, as well as data and function bindings. Also included is a prototype for rule *interlingua*, which is currently based on CLP; later, it will be based on *ruleML* (the proposed standard rule format) in order to enable interoperability of different rules.

CommonRules is being piloted in EECOMS, a $29 million, three-year industry consortium development effort on inter-enterprise supply chain integration for manufacturing.

How does it work?

CommonRules provides innovative XML interoperability and prioritized conflict-handling capabilities. These modularly augment a wide variety of rule-based systems and programming mechanisms already available in the market. CommonRules 3.3 includes an API set for enhancing Java or non-Java applications. It also includes extensive documentation and example rule sets.

Using CommonRules, a seller Web site or application can communicate in XML its business policy rules about pricing, promotions, customer service provisions for refunds and cancellation, ordering lead time, and other contractual terms and conditions, to a customer application or agent, even when

About this tech

- Overview
- Requirements
- FAQs
- Download!
- Discussion Forum
- About the Authors
- Press Articles
- Evaluate this Tech

Development F
DB2 develope
WebSphere d resources
Tivoli develop
Lotus develop
Rational devel resources

New Licensable T
J323 Engine
IBM DLPAR T pSeries
Web Performa
High Resolutic Stamp Facility

Top Online
Emerging Tec Toolkit
Simple Brows Productivity C

Top Down
IBM Toolkit fo
Structural Ana Java
Design Patter
JAR Class Fin
HeapRoots

the seller's rules are implemented using a different rule system (such as OPS5-style production rules) than that in which the buyer's rules are implemented (such as Prolog). The customer application or agent can then understand and assimilate those rules into its own business logic, and it can automatically execute those rules to make plans or decisions.

Trademarks:
Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc., in the United States, other countries, or both.

Platforms:	Categories:
All Java Platforms; Windows 95; Windows NT; Windows 98	appdev - Architecture java - APIs

CommonRules Discussion
Post New Topic • **Search Forum**

EJB Compliant? by Glenn Williams on 12/04/2003 10:51:41 AM

...more Postings

alphaWorks Conditions of Use

About IBM | Privacy | Terms of use | Contact

Hit List

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Search Results - Record(s) 1 through 11 of 11 returned.

☐ 1. Document ID: US 20040064351 A1

Using default format because multiple data bases are involved.

L19: Entry 1 of 11 File: PGPB Apr 1, 2004

PGPUB-DOCUMENT-NUMBER: 20040064351
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20040064351 A1

TITLE: Increased visibility during order management in a network-based supply chain environment

PUBLICATION-DATE: April 1, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Mikurak, Michael G.	Gulfport	FL	US	

US-CL-CURRENT: 705/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 2. Document ID: US 20030033179 A1

L19: Entry 2 of 11 File: PGPB Feb 13, 2003

PGPUB-DOCUMENT-NUMBER: 20030033179
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20030033179 A1

TITLE: Method for generating customized alerts related to the procurement, sourcing, strategic sourcing and/or sale of one or more items by an enterprise

PUBLICATION-DATE: February 13, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Katz, Steven Bruce	Burlingame	CA	US	
Labrou, Yannis	Menlo Park	CA	US	
Thompson, Jonathan P.	Redwood City	CA	US	
Rudin, Kenneth M.	San Carlos	CA	US	

US-CL-CURRENT: 705/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 3. Document ID: US 20020178271 A1

L19: Entry 3 of 11 File: PGPB Nov 28, 2002

PGPUB-DOCUMENT-NUMBER: 20020178271
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020178271 A1

TITLE: Dynamic file access control and management

PUBLICATION-DATE: November 28, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Graham, Todd D.	Ann Arbor	MI	US	
Hudson, Jonathan C.	Grosse Pointe Farms	MI	US	

US-CL-CURRENT: 709/229; 705/57, 705/59

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 4. Document ID: US 20020178077 A1

L19: Entry 4 of 11 File: PGPB Nov 28, 2002

PGPUB-DOCUMENT-NUMBER: 20020178077
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020178077 A1

TITLE: Method for automatically invoking a software module in response to an internal or external event affecting the procurement of an item

PUBLICATION-DATE: November 28, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Katz, Steven Bruce	Burlingame	CA	US	
Labrou, Yannis	Menlo Park	CA	US	

US-CL-CURRENT: 705/26; 705/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 5. Document ID: US 20020174000 A1

L19: Entry 5 of 11 File: PGPB Nov 21, 2002

PGPUB-DOCUMENT-NUMBER: 20020174000
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020174000 A1

TITLE: Method for managing a workflow process that assists users in procurement, sourcing, and decision-support for strategic sourcing

PUBLICATION-DATE: November 21, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Katz, Steven Bruce	Burlingame	CA	US	
Labrou, Yannis	Menlo Park	CA	US	
Kanthanathan, Manoranjan	Foster City	CA	US	
Rudin, Kenneth M.	San Carlos	CA	US	

US-CL-CURRENT: 705/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 6. Document ID: US 20020138316 A1

L19: Entry 6 of 11 File: PGPB Sep 26, 2002

PGPUB-DOCUMENT-NUMBER: 20020138316
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020138316 A1

TITLE: Value chain intelligence system and methods

PUBLICATION-DATE: September 26, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Katz, Steven Bruce	Burlingame	CA	US	
Labrou, Yannis	Menlo Park	CA	US	
Dam, Truc T.	San Jose	CA	US	
Dickerson, Joyce Leah	Menlo Park	CA	US	
Kanthanathan, Manoranjan	Foster City	CA	US	
Kolodney, David	San Jose	CA	US	
Levine, Timothy Edge	Sunnyvale	CA	US	
Li, Richard C.	Menlo Park	CA	US	
Nori, Ram Kumar	Sunnyvale	CA	US	
O Neill, Eamon Gearoid	San Francisco	CA	US	
Rudin, Kenneth M.	San Carlos	CA	US	
Tran, Quoc Tai	Mountain View	CA	US	

US-CL-CURRENT: 705/7

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 7. Document ID: US 20020065745 A1

L19: Entry 7 of 11 File: PGPB May 30, 2002

PGPUB-DOCUMENT-NUMBER: 20020065745
PGPUB-FILING-TYPE: new
DOCUMENT-IDENTIFIER: US 20020065745 A1

TITLE: Rule-based personalization framework for integrating recommendation systems

PUBLICATION-DATE: May 30, 2002

INVENTOR-INFORMATION:

NAME	CITY	STATE	COUNTRY	RULE-47
Rainsberger, Joseph B.	Toronto		CA	
Tin, Ramiah Kwok-Fai	Pickering		CA	
Tong, Tack	Scarborough		CA	

US-CL-CURRENT: 705/27; 706/47

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☑ 8. Document ID: US 6697824 B1

L19: Entry 8 of 11 File: USPT Feb 24, 2004

US-PAT-NO: 6697824
DOCUMENT-IDENTIFIER: US 6697824 B1

TITLE: Relationship management in an E-commerce application framework

DATE-ISSUED: February 24, 2004

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Bowman-Amuah; Michel K.	Colorado Springs	CO		

US-CL-CURRENT: 709/229; 709/223

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw D

☐ 9. Document ID: US 6671818 B1

L19: Entry 9 of 11 File: USPT Dec 30, 2003

US-PAT-NO: 6671818
DOCUMENT-IDENTIFIER: US 6671818 B1

**** See image for Certificate of Correction ****

TITLE: Problem isolation through translating and filtering events into a standard object format in a network based supply chain

DATE-ISSUED: December 30, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Mikurak; Michael G.	Hamilton	NJ		

US-CL-CURRENT: 714/4; 714/43, 714/48

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 10. Document ID: US 6629081 B1

L19: Entry 10 of 11 File: USPT Sep 30, 2003

US-PAT-NO: 6629081
DOCUMENT-IDENTIFIER: US 6629081 B1
**** See image for Certificate of Correction ****

TITLE: Account settlement and financing in an e-commerce environment

DATE-ISSUED: September 30, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Cornelius; Richard D.	Santa Monica	CA		
Stepniczka; Andreas	San Francisco	CA		
Chu; Kevin	Atlanta	GA		

US-CL-CURRENT: 705/30

Full | Title | Citation | Front | Review | Classification | Date | Reference | Sequences | Attachments | Claims | KWIC | Draw De

---

☐ 11. Document ID: US 6606744 B1

L19: Entry 11 of 11 File: USPT Aug 12, 2003

US-PAT-NO: 6606744
DOCUMENT-IDENTIFIER: US 6606744 B1

TITLE: Providing collaborative installation management in a network-based supply chain environment

DATE-ISSUED: August 12, 2003

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Mikurak; Michael G.	Hamilton	NJ		

US-CL-CURRENT: 717/174; 705/26, 717/178

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw D

Clear | Generate Collection | Print | Fwd Refs | Bkwd Refs | Generate OACS

Terms	Documents
L17 and rule adj engine	11

Display Format: - Change Format

Previous Page Next Page Go to Doc#

NPL Services for Examiners

NEW ScienceDirect Journals

Thursday, April 1, 2004

STIC's mission is to connect examiners to critical prior art by providing information services and access to NPL electronic resources and print collections. A STIC facility is located in each Technology Center.

Most of the electronic resources listed on these Web pages are accessed via the Internet. You must be authenticated for data to be accessed. ▶ Firewall Authentication

Specialized Information Resources for Technology Centers

Select a Technology Center

Technology Centers: GO

Information Resources and Services

- Breaking News on Emerging Technologies
- List of Major E-Resources
- List of eBook and eJournal Titles
- Reference Tools
- Legal Resources
- Nanotechnology
- STIC Online Catalog
- PLUS System
- Foreign Patent Services
- Translation Services
- Trademark Law Library

Request STIC Services from your Desktop

- Request a Prior Art Search
- Request Delivery of Article or Book Reference
- Request Purchase of a Book/Journal
- Request Foreign Patent Document
- Request a Translation

Request PLUS Search

Last Modified: 03/25/2004 12:36:15

TC2100: EIC Resources and Services





ScienceDirect Journals
Daily Breaking News on Emerging Technologies:
- Encryption
- Information & Data Security
- Internet Security

Thursday, April 1, 2004

These resources and services provide examiners with access to critical prior art. Most of the electronic resources listed on these Web pages are accessed via the Internet. You must be authenticated for data to be accessed. ▶ Firewall Authentication

➪ indicates tools featured in TC's NPL training.

Information Resources

Information Resources by Class and Subclass

Databases

➪ ACM Digital Library
Business Source Corporate
(Multidisciplinary subject coverage)
Dialog Classic on the Web
(Training and password required.)
DTIC STINET
(Citations of Defense Technical Information Center scientific and technical documents)
EEDD Submission Form
Examiners' Electronic Digest Database (EEDD)
(Database of examiner submitted NPL)
EPOQUE
(EPO's databases, available on stand-alone terminal in CPK2, 4B40)

➪ IEEE Xplore
(Full page images of over 800,000 Electrical & Electronic Engineering articles, papers and standards, 1988 - present. Select content is available from 1952-1987.)
INSPEC
(Seven million well-indexed physics, EE, and IT abstracts, 1969-present)
IP.com
(Defensive disclosures published to the Disclosures IP.com database from various websites)
NTIS (National Technical Information Service)
(resource for government-funded scientific, technical, engineering, and business related information)
Proquest Direct
(Multidisciplinary subject coverage)
Readers' Guide to Periodical Literature
(citations to popular multidisciplinary magazines)
Research Disclosure
(Published monthly as a paper journal and now as an online database product with advanced full text searching capabilities for defensive disclosure information.)

Software Patent Institute (SPI) (Select "Free Access")
(Searchable database of Software Technologies.)
SPIE Digital Library
(journals and proceedings on optics and photonics)
STN on the Web (training and password required)
(The other link is via the Patent Examiner's Toolkit. On your computer, click on the START button, then on the PE Toolkit, then on STN Express.)
True Query
(A resurrected version of the old "Computer Select" database, providing full text access to over 100 technology focused publications, a glossary of technical terms, product reviews and over 60,000 product specifications from 1999 to the present. If html code appears on your screen, click browser's "Reload" or "Refresh" button.)

Books and Journals

Search STIC Online Catalog
InfoSECURITYnetBASE
(Information security)
Knovel
(Applied science and engineering)
NetLibrary.com
(Multidisciplinary subject coverage)
Safari Online Books
(Computer and information technology)
ScienceDirect
(scientific, technical, and medical journals)
Springer Publishing Company
(biotech, physics, and computer journals)

Daily Newspapers

Fulltext newspaper articles are available electronically in Proquest Direct.

CD-ROM Resources

Older full text NPL resources/articles received in CD-Rom format. These resources are available on EIC2100 PCs in CPK2, 4B40.

Equipment

Reference Tools

Bartleby.com
(Several versions of Roget's Thesaurus, a dictionary, an encyclopedia, quotations, English usage books and more.)
Computer References
(Dictionaries, Acronyms Finders, Encyclopedias)
Efunda
(30,000 pages of engineering fundamentals and calculators)
Encyclopedia Britannica
Encyclopedia of Software Engineering
Eric Weisstein's World of Mathematics
(A comprehensive online encyclopedia of mathematics.)
HowStuffWorks
(Search a term to find articles that explain how it works.)
Over 2000 Glossary Links
(Links to numerous technical, specialty, and general glossaries.)
PCWebopedia
Wiley Encyclopedia of Electrical and Electronics Engineering
Yourdictionary.com

(Numerous "specialty dictionaries"... technological, law, business related and more.)

Services

EIC2100 Staff
Foreign Patent Services
PLUS
Request a Book/Journal Purchase
Request a Book or Article
Request a Foreign Patent Publication
[e-submit] [Printable form]
Request a Prior Art Search
[e-submit] [Printable form]
Fast & Focused Search Criteria
STIC Online Catalog
Translation Services

Web Resources

A Brief History of the Hard Disk Drive

CiteSeer (ResearchIndex)
(Full text scientific research papers - in pdf and postscript formats.)

Internet Engineering Task Force
(The IETF Secretariat, run by The Corporation for National Research Initiatives with funding from the US government, maintains an index of Internet-Drafts.)

Nanotechnology

Requests for Comments (RFCs) Database
(Requests for Comments (RFC) document series is a set of technical and organizational notes about the Internet (originally the ARPANET), beginning in 1969 and discussing many aspects of computer networking, including protocols, procedures and concepts as well as meeting notes and opinions.)

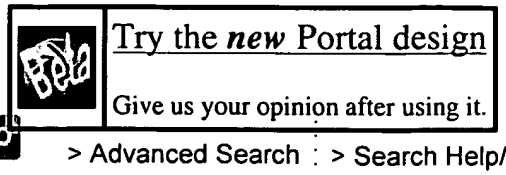
Usenet Archive (Google Groups)

Wayback Machine
(Archived web pages.)

Submit comments and suggestions to Anne Hendrickson

To report technical problems, click here





ACM Digital Library

A half century of pioneering concepts and fundamental research have been digitized and indexed in a variety of ways in this special collection of works published by ACM since its inception. The ACM Digital Library includes bibliographic information, abstracts, reviews, and full texts.

Digital Library Overview
- **What's New**
- **FAQ**
- **DL Pearls**
- **Content and Organization**
- **Terms of Usage**
- **Resources from Affiliated Organizations**

Subscription and Access Information

> Access Information
> Individual Subscriptions
> Institutional Subscriptions

Browse the Digital Library

- **Journals**
- **Magazines**
- **Transactions**
- **Proceedings**
- **Newsletters**
- **Publications by Affiliated Organizations**

- **Special Interest Groups (SIGs)**

Personalized Services

My Bookshelf — Custom collections. Personal virtual Journals. Intelligent agent searches. Collaborative filtering.

Online Computing Reviews Service

OCRS — Access critical reviews of the computing literature using the Online Computing Reviews Service.

> Document Delivery Service

The ACM Digital Library is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

Read the ACM Privacy Policy and Code of Ethics
Questions? Comments? Contact webmaster@acm.org
Call: 1.800.342.6626 (USA & Canada) or +212.626.0500 (Global)
Write: ACM, 1515 Broadway, New York, NY 10036, USA

Personalizing E-commerce Applications with On-line Heuristic Decision Making

Vinod Anupam Richard Hull Bharat Kumar
Bell Labs, Lucent Technologies
600 Mountain Avenue
Murray Hill, NJ 07974
{anupam,hull,bharat}@lucent.com

ABSTRACT

This paper describes new technology based on on-line decision support for providing personalized customer treatments in web-based storefronts and information sites. The central improvement over existing systems is a new paradigm for specifying decisions, based on a language that incorporates flowchart constructs, rules-based constructs, and a variety of specialized constructs to facilitate reasoning based on heuristics and partial information. Reports about decisions made by a program in this language have structure that is conceptually close to the structure of that program. This makes it easy for business analysts and managers to tune the programs to enhance business performance.

To illustrate the benefits of our approach this paper describes the May-I-Help-You (MIHU) prototype system, that monitors a customer's progress through a web storefront, and may choose to proactively intervene in order to help close a sale. The intervention might offer a discount or promotion, or give the customer a "May I Help You" window, that offers an opportunity to have text chat, voice chat, and/or escorted browsing with a Customer Service Representative (CSR). In MIHU, the decision about whether to offer live assistance is fully automated, taking into account not only the business value of a given customer interaction, but also the current availability of CSRs to help realize this opportunity.

Keywords

B2C E-commerce, personalization, pro-active intervention, Vortex rules system

1. INTRODUCTION

The advent of e-commerce is forcing radical changes to the landscape of marketing and customer care. Customers are demanding increased flexibility and convenience in accessing information about products, in ordering them, and obtaining service for them. At the same time, businesses are attempting to support (a) "segment of one" marketing and service to large masses of people [18, 20], including intelligent targeted advertising, and intelligent mechanisms to identify and take advantage of profitable and loyal customers; and (b) meaningful dialogues with customers so that quality of service can be improved before customers switch to a competitor. These needs are not restricted to B2C e-commerce; web-sites in B2B e-commerce that are accessed by employees of a business must also provide effective, personalized service. This paper introduces a new approach to personalizing web-based e-commerce sites called DFP (Decision Flow Personalization), that is based on the use of on-line decision support. A central contribution is the use of a novel language for specifying decisions, that supports flowchart constructs and a specialized construct called "Decision Flow", that combines rules-based constructs and a variety of specialized constructs to facilitate reasoning based on both heuristics and partial information. The DFP approach is illustrated here by describing the MIHU (May-I-Help-You) prototype system, that proactively offers live assistance to web storefront customers.

A fundamental challenge in supporting personalization through on-line decision support is to create a high-level language for specifying decisions that supports sophisticated reasoning but which at the same time is accessible to business analysts and managers. As a starting point for our work, we interviewed business experts on customer care and personalization to understand the features that they need from a decision support language. The following requirements were determined:

(a) Ability to use both formal (e.g., chaining of rules) and heuristic (e.g., giving scores based on *ad hoc* combinations of various factors) styles of reasoning;

(b) Ability to use rules where appropriate, and to use flowchart constructs where appropriate;

(c) Ability to work with partial and/or incomplete information;

(d) Possibility for hierarchical, modular structuring;

(e) Ability to bring in outside information (e.g., access to customer profiles, the results of bulk statistical analysis);

(f) Ability to invoke side-effect functions (e.g., database updates, triggering workflows).

Copyright is held by the author/owner.
WWW10 May 1-5, 2001, Hong Kong.
ACM 1-58113-348-0/01/0005.

(g) A clear and intuitively natural semantics;

(h) A natural correspondence between reports on decisions made and the structure of how the decisions are specified (i.e., primarily the structure of the rule sets); and

(i) The language can be "owned" or controlled by business analysts and managers, without relying on programmers that translate the decision specifications into a highly technical format.

Some additional systems requirements were determined:

(j) The on-line decision engine should permit changes to decision specifications with no interruption in service; and

(k) User-friendly authoring of decision policies, including rules.

As detailed in Section 5, the rules-based decision specification languages used in existing approaches for e-commerce personalization (e.g., Manna [15], Blaze [2]) satisfy some but not all of these requirements because of their limited expressive power, and other approaches to decision specification (e.g., expert systems, logic programming) fail to satisfy some of the requirements because they are too rich.

To fill this void we use a new paradigm for specifying decisions, called Vortex. An early version of this paradigm is described in [12], where the focus was on the flexible specification of workflows that incorporated business heuristics. Central to the Vortex paradigm is the notion of "Decision Flow", which is a novel combination of rules constructs and workflow-like constructs. As detailed in Section 3, in a Decision Flow the emphasis is on computing attribute values. Some of these are targets of the decision (e.g., should a discount be offered to a customer) and others are intermediate to the decision (e.g., the likelihood that this customer will leave the site before completing the deal). Rules may be used to compute the values of individual attributes, and rules may be used to control what attributes are to be computed. (For example, attributes not relevant to a specific decision can be ignored.) Reports about decisions made can show the values of the target and intermediate attributes, and have structure close to the structure of the Decision Flow.

Decision Flows permit complex reasoning about a broad array of data about web sessions and customers. To take full advantage of this it is important to have access to rich information about the pages a customer is visiting, including the underlying intent of the pages (e.g., is it a catalog page, an instructions page, a shopping cart page) and the content delivered in them (e.g., what is the quality of a search result). Section 4 outlines and compares different approaches for gaining access to that information.

To illustrate the core technology and benefits of DFP this paper introduces the May-I-Help-You (MIHU) prototype system. This system is aimed at reducing the number of abandoned e-commerce transactions. Industry statistics [7] indicate that in the U.S. market, for every online B2C transaction that is completed there are nearly four times as many that are abandoned. Further, 7.8% of the abandoned transactions could be converted into sales by using live Customer Service Representative (CSR) interaction. This translated into $6.1 billion in lost e-commerce sales in 1999, and could lead to a cumulative loss of more than $173 billion in the subsequent 5-year period.

The MIHU system monitors a customer's progress through a web storefront, and uses stored and real-time information to infer values such as the current business value of the session and the frustration level of the customer. The MIHU system can proactively offer the customer discounts or targeted promotions. Further, the MIHU system can offer the customer a "May I Help You" window, which invites the customer to interact with a Customer Service Representative (CSR), through text chat, voice chat, and/or escorted web browsing. The decision server accesses both stored information and real-time information, including the current availability of, and load on, the CSRs.

Lucent Technologies is developing a product, called Contact Assist, that will support the functionality of the MIHU prototype system, including both an engine based on Vortex and a flexible mechanism for gathering information from a web server. Contact Assist will be available in mid 2001.

The DFP approach can be used in a wide variety of e-commerce applications involving personalization and customization, including the offering of carefully targeted promotions and discounts, helping with navigation through catalogs or self-help material, guiding a customer through an ordering process, and conducting automated dialogues with the customer. It can also be used in non-commercial web-based applications, including context-aware searching tools and automated customization of portals.

Organization. As noted above, the MIHU system will be used to illustrate the main features of our approach. For this reason, we begin in Section 2 by describing the MIHU system at a high level. Section 3 describes the Vortex and Decision Flow paradigms and illustrates their use in connection with the MIHU system. Section 4 describes approaches for incorporating on-line decision servers, such as Vortex, into web-based e-commerce sites. Section 5 considers related work. Section 6 discusses future research directions.

2. EXAMPLE APPLICATION: MAY I HELP YOU

In a department store, customers are free to browse. In a good department store, a salesperson will sometimes approach customers with the gentle question "May I help you?". In an excellent department store the timing and manner in which this question is asked is guided largely by the browsing behavior of the customer. The May-I-Help-You (MIHU) system provides a functionality for web-based storefronts that is analogous to this kind of service in excellent department stores. The MIHU system has an important advantage over a department store salesperson, which is that many businesses know the identity of customers during their visits on the web.

MIHU is a Customer Relationship Management system that interfaces to a business' web storefront. MIHU can keep track the interaction of a customer with the storefront. To be more specific, using the high-level Vortex language business analysts and managers can program the MIHU system to use customer interaction information (e.g., shopping cart content, sequence of pages visited), coupled with information available in enterprise databases (e.g. customer profile, contact history, current orders, and results from off-line decision support tools) to build a model of the customer and the

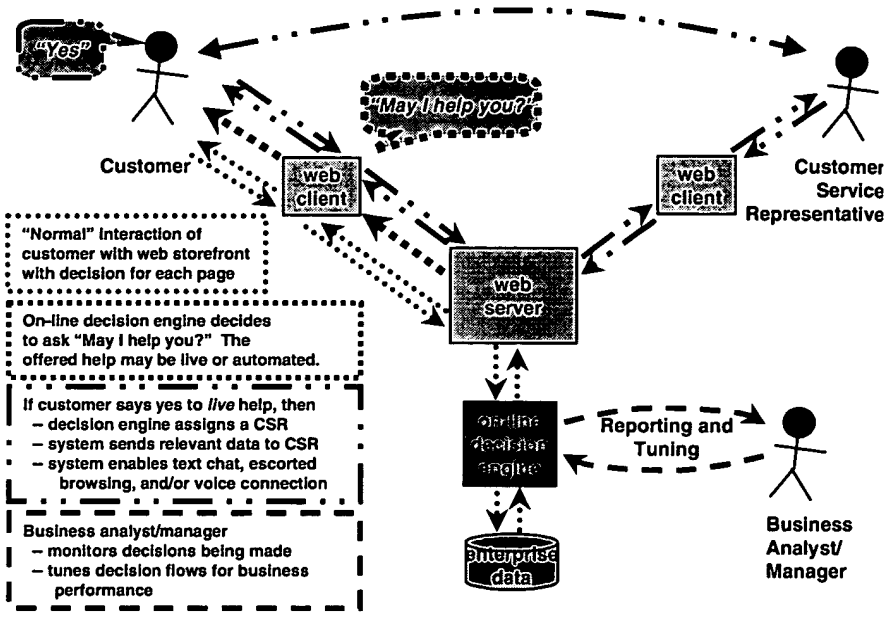


Figure 1: Overview of MIHU functionality

current interaction. Based on the individual characteristics of each customer interaction, MIHU may choose to present to the customer an icon or window offering help relevant to the current context. This help might be automated, or it might be an offer to chat with a live Customer Service Representative (CSR). In the first case, if the customer takes up the offer (e.g., by clicking on the icon or window), then appropriate context-dependent information will be delivered to the customer. In the second case, a CSR will be assigned, appropriate information will be forwarded to that CSR, and some kind of interaction with the customer will be initiated. Of course, providing live CSR help with an interactive session brings with it the opportunity to help close the sale, and also the opportunity to attempt cross-sells or up-sells.

Figure 1 summarizes the operation of the MIHU system. There are four phases or aspects to the operation. In the first phase (shown with lines having small dots) a customer has "normal" interaction with the web store-front. In particular, the web server supporting the store-front presents pages to the customer's web client, and the customer fills in blanks and submits page requests to the server. However, before the web server presents a new page to the customer, the on-line decision engine is asked whether or not the customer should be presented with the "May I Help You" option (or some other optional assistance or customer service such as a targeted discount). The decision server can access information about the customer's current web session (e.g., pages visited, shopping cart contents), and may access data from an enterprise database (including results from off-line decision support systems). The decision server may also gather information from decision engines using alternate reasoning paradigms, such as an expert system or, e.g., a specialized system for determining customer preferences.

The second phase (shown with lines having large dots) occurs if and when the decision engine determines that the customer should be given the MIHU option. In that case, the web server presents to the customer's client an applet that asks whether the customer would like assistance from a CSR.

The third phase (shown with lines having short and long dashes) arises if the customer does want assistance. In that case, some or all of three forms of interaction can be established between customer and CSR: voice conversation, text chat, and "escorted" or "collaborative" web browsing (where the CSR can select a URL and both the CSR and customer clients go to that URL, or visa-versa).

The fourth phase (shown with lines long dashes) occurs in parallel with the other ones, and at a more deliberate pace. This stage involves *tuning for business performance*, i.e., the continued examination of the decisions made for the web-storefront, with the ultimate goal of making improvements on the underlying decision policies. As will be described below, novel aspects of the Vortex language make it possible to quickly modify a Vortex program in order to achieve a desired effect.

At a superficial level, it might seem that since the MIHU system monitors a customer's progress through a web site, and peeks at the interaction between her and the web site, there are serious privacy issues involved here. However, this is not the case since the MIHU system is not getting any extra information that is not already available to the web site. However, it might be a good idea to let the customer know that such monitoring might be going on (e.g., by allowing her to opt-in when she registers with the site).

3. VORTEX AND DECISION FLOWS

This section presents a detailed introduction the Vortex language using the MIHU example, indicates how Vortex satisfies the requirements given in the Introduction, and describes the engine for executing Vortex programs. Some formal details about the Vortex language are beyond the scope of this paper, but may be found in [12].

In the current version of Vortex, programs are essentially flowcharts that may include one or more specialized nodes

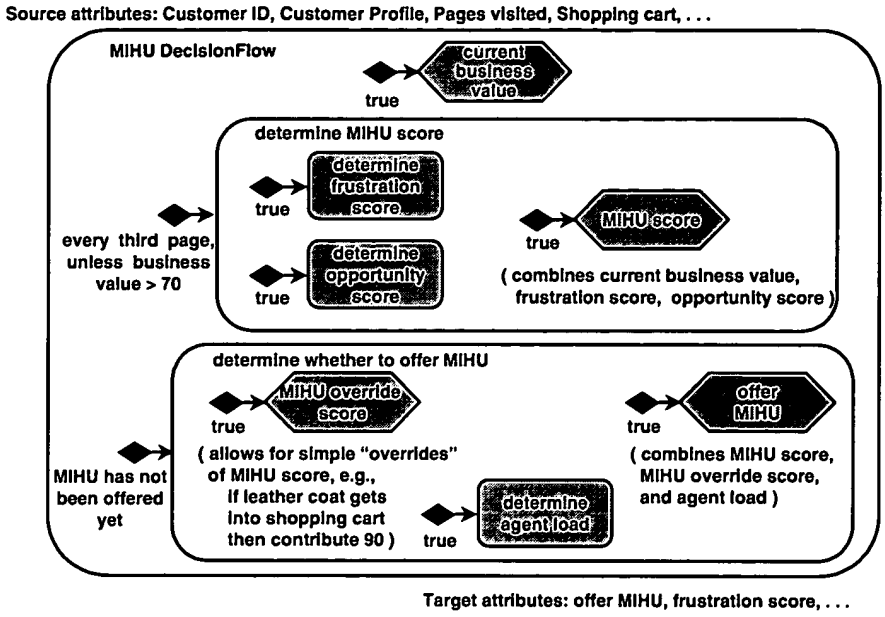


Figure 2: Representative Decision Flow for making MIHU decision (using informal syntax)

which contain "Decision Flows". Since flowchart constructs are well understood, we focus here on Decision Flows.

3.1 Vortex Decision Flows

The Decision Flow paradigm was developed for specifying complex reasoning that may involve partial information, heuristics, and multiple styles of combining information. A key criterion in the development of the paradigm was that users other than computer scientists (e.g., business managers, policy analysts, domain experts) should be able to understand the specifications of how decisions will be made, and in some cases be able to modify the specifications directly. Decision Flows support a form of incremental decision-making, that can easily incorporate a myriad of business and other factors, and specify the relative weights they should be given. Decision Flows support a rule-based style of specifying decision policies, and are more expressive than decision trees and traditional business rule systems. However, Decision Flows are less expressive than conventional expert systems as a result of a novel approach to structuring the rule set underlying a Decision Flow. This helps to simplify explanations of how a decision is made, and reduces the "ripple effect" that often arises from modifications to programs written in with expert systems or logic programming languages.

We briefly illustrate some of the basic Decision Flow constructs using Figures 2 and 4, which give a high-level picture of a representative Decision Flow that can be used for making the MIHU decision. In the MIHU prototype, a Decision Flow like this would be one node of a flowchart which is executed for each page that is served to a customer. For example, this outer flowchart might operate as follows: (a) test whether the page indicates a new session or is the continuation of an active session, (b) if a continuation then retrieve information (from a main memory database or internal data structure) about previous pages of the session (c) possibly get customer profile information, (d) execute a Decision Flow that decides whether to make an automated intervention, and (e) inform the web server about the decision made.

The two Decision Flow figures show rules and conditions using an informal, pidgin syntax. In the text-based version of Vortex, the syntax for conditions and terms is close to the C language. A GUI is provided for Vortex programmers, including wizards to help with rule construction, query construction, and the like.

Decision Flows are *attribute-centric*. In particular, a Decision Flow specification has *source attributes* or input parameters; in the example these hold information about the customer identification, customer profile and current session. The specification also includes a family of *derived attributes*, which may be evaluated during execution. Some of the derived attributes will be *target*, and embody the output of a Decision Flow; in the example this includes a boolean indicating whether to offer the MIHU functionality, and additional attributes giving characteristics of a session. The current prototype Vortex system supports data types associated with relational databases, namely scalars, tuples of scalars, lists of scalars, and lists of tuples of scalars. (We expect to incorporate XML data in the next round.)

The Decision Flow of Figure 2 shows individual attributes using hexagons (e.g., `current business score`, `offer MIHU`), and modules using rounded-corner boxes (e.g., `determine MIHU score`, `determine frustration score`). A hexagon node may contain rules that specify how an attribute is to be computed; this will be described below in Subsection 3.2. External functions such as database queries, calls to a heavy-weight decision support system (e.g., an expert system), or side-effect functions (e.g., database updates, triggering workflows) can also be included. The modules may be hierarchically organized, and may contain other modules, hexagons, and external functions.

We now use Figure 3 to explain intuitively how the MIHU Decision Flow operates. This figure shows a report present-

Session ID	Page	Business Value	Frustration Score	Opportunity Score	MIHU score	Override Score	CSR Load	Offer MIHU
250	1	70	6	0	46	0	80	false
250	2	70	14	0	54	0	81	false
250	3	70	20	0	60	0	81	false
250	4	70	24	0	64	0	79	false
250	5	85	8	51	83	0	78	true
282	1	40	6	0	25	0	83	false
282	2	40	–	–	–	0	82	false
282	3	57	–	–	–	90	82	true
282	4	57	16	29	43	–	–	–
282	5	57	–	–	–	–	–	–
282	6	71	19	37	58	–	–	–
282	7	71	25	48	58	–	–	–

Figure 3: Data from report on MIHU decisions

ing some representative decisions reached by the decision engine. The columns of this report correspond to some of the most important attributes of the Decision Flow, and each row corresponds to a single execution of the Decision Flow. During a single execution, the value of `offer MIHU` is based on three intermediate attributes: `MIHU score`, `MIHU override score`, and `CSR load`. In the example, if either of the scores is $\geq$ `CSR load`, then the MIHU functionality is offered.

The `MIHU score` attribute is based on other intermediate attributes, which focus on the current business value of the customer and session, on the estimated frustration level of the customer, and on the estimated opportunity for making money from the customer (either by encouraging the customer to purchase the contents of the shopping cart, or through a cross-sell or up-sell). The frustration score and opportunity scores in turn depend on additional intermediate attributes.

Referring to Figure 3, the first five rows of Figure 3 show how the different scoring attributes might vary over a user session. We assume in this example that the customer visited 5 pages, and placed something in the shopping cart when sending the 4th page back to the web storefront. In the Decision Flow used to generate this example, `frustration score` goes up, except when the customer places something in the shopping cart. The intuition here is that customer frustration goes down if there is a feeling of progress, e.g., after several searches a product is found and put in the shopping cart. On the other hand, the `opportunity score` generally goes up when something goes into the shopping cart, both because there is something in the shopping cart, and in some cases there are possibilities for cross-sells and up-sells.

Of course, the specific behavior of the attributes in a Decision Flow is determined by the business analysts and managers who program it. As a result, the Decision Flow for MIHU described here can be adapted to encompass any principles and heuristics that a business manager wants.

A key feature of Decision Flows is the use of *enabling conditions* or guards on the execution of attributes, modules, and external functions. For example, `determine MIHU score` has as enabling condition, expressed informally, that the module will be executed if the `current business value` is > 70, and otherwise on every third page of the web session. Likewise, `determine whether to offer MIHU` will be executed only if the MIHU option has not yet been offered to the customer.

Session 282 in Figure 3 illustrates how the enabling condition on `determine MIHU score` impacts Decision Flow executions. In this session the `current business value` is ≤ 70, and so the `determine MIHU score` is not executed on the 2nd, 3rd or 5th pages. On the 6th page the `current business value` goes above 70. So `determine MIHU score` is computed for the 6th and 7th page.

Enabling conditions are useful in at least three contexts: (a) to permit savings on resource usage (as just illustrated); (b) to avoid the computation of irrelevant attributes (e.g., once MIHU has been offered there is no need to compute `offer MIHU`); and (c) to indicate which attributes should be ignored if a realtime constraint is about to be violated.

What happens if an attribute that has been disabled is referred to by the computation of some other attribute? One design principle of Decision Flows is that any attribute may have null value, and that any attribute computation must be able to work with null inputs. This is motivated in part by the observation that data retrieval over a network is not reliable, and that many decisions must be made using partial information. Suppose in the example that the MIHU option has not been offered yet. Even if the module `determine MIHU score` is disabled, i.e., not evaluated, the rest of the Decision Flow will be executed, and a final value for `offer MIHU` will be obtained. The condition language used in Decision Flows can test whether an attribute has been disabled (i.e., the enabling condition is false). Importantly, both the condition language and the attribute computation language used in Decision Flows were designed to work in the context of partial information and null values (see [12]).

As detailed in [12], a declarative semantics is associated with Decision Flows. Under this semantics Decision Flows are viewed as input-output devices, which map a given set of source attribute values (and an underlying environment, such as any databases accessed) to a given set of target attribute values. It turns out that given a set of source attribute values (and a fixed underlying environment), a well-formed Decision Flow uniquely determines the values of the target attributes. A key factor in achieving this declarative semantics is that Decision Flows must satisfy a certain

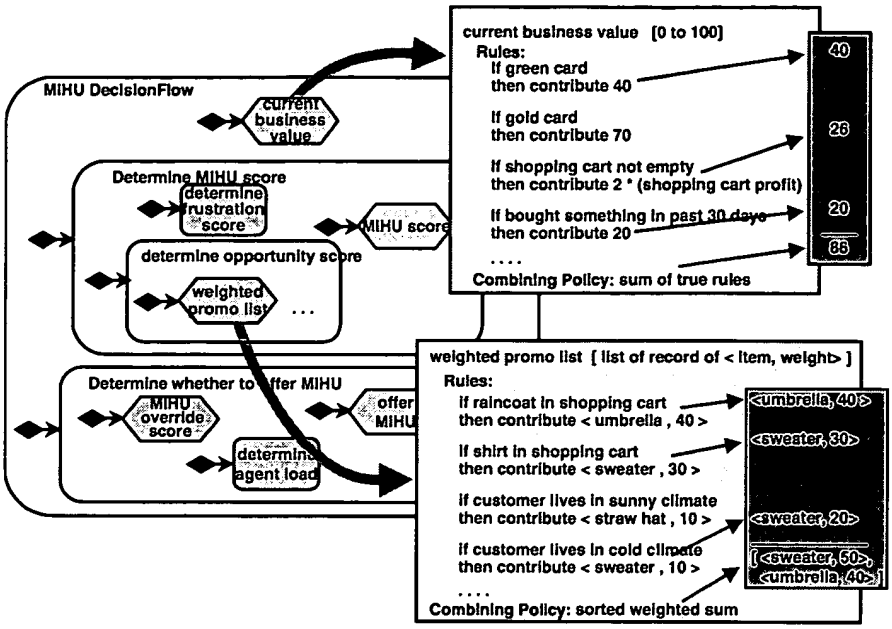


Figure 4: Attribute Rules and Combining Policies in a Decision Flow

acyclicity property. In particular, a graph can be formed for each module M, where the nodes are the top layer modules, attributes, and external functions of M, and which contains an edge from node A to node B if (i) [*data flow*] an attribute defined in A is used in the computation of B, or (ii) [*enabling flow*] an attribute defined in A is used in the enabling condition of B. For a Decision Flow to be well-formed, this graph must be acyclic for each module. In operational terms, the acyclicity condition implies that there will not be race conditions between different attributes being evaluated. Further, the acyclicity condition underlies our claim that Decision Flows are easier to understand than expert systems, and suffer less from the ripple effect.

There are three advantages to the declarative semantics just outlined. First, the semantics provides a clear and unambiguous meaning for Vortex Decision Flows. Second, people developing Vortex Decision Flows can largely ignore flow of control issues, and focus instead on the business logic they are trying to express. This provides a key difference between Decision Flows and conventional flowcharts. (The Vortex compiler will alert the user if the acyclicity condition is violated.) And third, the declarative semantics affords some possibilities for optimizations, of both response time and system throughput (see [11]).

3.2 Attribute Rules and Combining Policies

Another key feature of the Decision Flow paradigm is the tremendous flexibility given to users when specifying how an attribute should be computed. In addition to permitting external function calls (e.g., database dips, or calls to execute in a different decision support engine) the paradigm supports the use of *attribute rules* and *combining policies*. Two simple illustrations are provided in Figure 4, which shows the contents of the `current business value` and `weighted promo list` attributes. As shown there, a family of rules is associated with attribute `current business value`, each potentially contributing a number. Numbers contributed by rules with true condition are to be combined by summation. In the example, rules contribute 40, 26 and 20, resulting in final value 86.

The attribute `weighted promo list` illustrates a more interesting combining policy. The output of this module will hold a list of promo items, ordered according to how well they fit the current situation. The individual rules contribute ordered pairs, consisting of a promo item along with a numeric weight (e.g., < umbrella, 40 >). As illustrated by the second and fourth rules here, several rules might contribute to the same promo item. The combining policy for this module is to group contributed pairs by promo item, then add the weights for each promo item, and finally sort the list of resulting pairs according to the aggregated weights.

More generally, the Decision Flow paradigm offers a broad range of combining policies for aggregating the contributions of a rule set. Other combining policies involving numbers include maximum, minimum and average. As illustrated with `weighted promo list` the contributed values and the result may have structured type. In addition to supporting a family of *ad hoc* combining policies, the system supports an OQL-like [4] algebra for specifying customized combining policies.

The presence of multiple combining policies permits the use of different styles of reasoning within the Decision Flow paradigm. Decision Flows also support different styles of reasoning at a more granular level as well. We illustrate this in connection with the attributes `MIHU score` and `MIHU override score`. We have discussed how `MIHU score` involves a deliberate derivation involving many factors. In contrast, the attribute `MIHU override score` is computed by an atomic node that includes collection of simple and disjoint factors (e.g., that a particular item is in the shopping cart, or that a certain page has been visited) and uses as combining policy "maximum contributed value". If `MIHU override score` is greater than `CSR load` then the MIHU

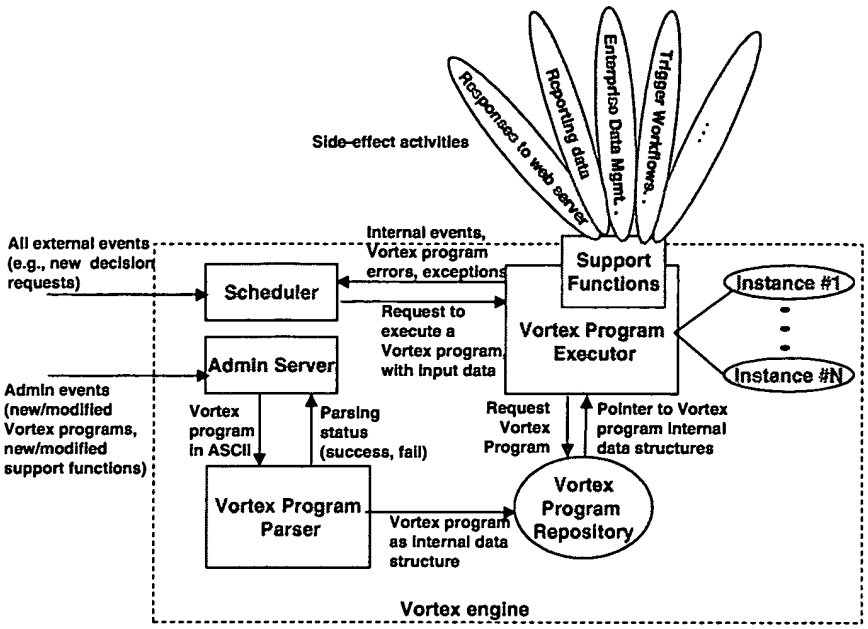


Figure 5: High-level architecture of Vortex engine

option will be offered, and so each rule in `MIHU override score` is analogous to a presidential veto or gubinatorial pardon. For example, in the second session of Figure 3 the `MIHU override score` goes to 90 on the 3rd page, perhaps because a leather coat was placed into the shopping cart; and this triggers the offer of MIHU.

3.3 Miscellaneous

This subsection gives added details about the Vortex system, and reviews it in light of the requirements given in the Introduction.

We have already seen from the preceding discussion that the Vortex language satisfies requirements (a) through (g).

Turning to requirement (h), the "attribute-centric" nature of Decision Flows makes possible reports about how or why decisions were made are conceptually close to the Decision Flow specification. In particular, reports such as in Figure 3 can be created using some or all of the attributes derived by the Decision Flow. Given a family of decisions, a user can inspect this report (either manually or using automated techniques such as regression analysis or data mining) to see whether the various attributes and criteria are given appropriate emphasis. If anomalies are found, then it is relatively easy to find the corresponding places in the Decision Flow that should be modified. Furthermore, we expect that this close correspondence between reports and Decision Flow structure will facilitate the development of self-learning tools that will work on top of Vortex.

Decision Flows reveal the key factors involved in making a decision or evaluation, and hide a substantial amount of detail about the execution. In contrast, when specifying an equivalent decision using a conventional flowchart or Petri Net formalism, the key factors and logic are obscured by the plumbing. In Decision Flows different ways of executing rules can co-exist; this contrasts with logic programming languages and conventional expert systems, which have a single execution semantics, and force the use of awkward simulations if rules are to be combined in a different way. It is these considerations along with the correspondence between reports and the structure of Decision Flows that lead us to believe that Vortex satisfies requirement (h).

We turn now to requirement (j). Figure 5 shows a high-level architecture of the Vortex engine. Vortex programs are input into the Administrative Server, which invokes a parser. This checks that the program is well-formed and compiles it into an internal data structure. When the program is to be executed, i.e., when a decision is to be made based on given input parameters, a copy of the data structure is created, and that copy is then interpreted. As a result, the Vortex program can be modified, parsed, and complied into a (new) internal data structure. The new data structure can then be used for subsequent decisions. In this way, Vortex programs can be modified without bringing the engine down. For efficiency, the Vortex engine has been implemented in C++. Furthermore, many of the specific operations of a Vortex program (e.g., arithmetic comparisons, list manipulations, external functions) are performed by "support functions", which are compiled. Additional support functions can be added to the engine without bringing it down.

With regards to requirement (k), a prototype GUI has been implemented to support specification of Vortex programs. A visual palette is provided for the Decision Flow constructs; this has appearance similar to the images of Figures 2 and 4. Wizards are provided for building up flowchart nodes, rules, attribute modules, database queries, etc.

The example MIHU Decision Flow described earlier is relatively simple, in terms of the size of the Decision Flow and the nature of the data being evaluated. We have developed richer Decision Flows that involve many modules and over 50 attributes.

4. INTEGRATION WITH WEB SERVERS

The second key component of the DFP approach to web personalization concerns creating a linkage between the on-line decision server and the web server hosting a site. We begin by discussing the kinds of information that need to be passed back and forth between web server and decision engine. We then consider different ways to convey relevant information to the decision engine, and to permit decisions made to have impact on the web server.

4.1 Raw vs. Semantic Information

The DFP approach to web personalization is based on providing relevant information to a sophisticated on-line decision engine. This subsection distinguishes between the raw data that can be obtained easily and higher-level semantic information, such as used by the Decision Flow in the example of Section 3.

At a minimum, the on-line decision engine should have access to the following kinds of information. We are not suggesting that all of this data will be used in each decision, but that it should be available if deemed relevant.

(a) *History of customer clicks*: This includes not only the web requests that the customer is making, but also the the navigation path being followed around the site, the times spent at each page, and the entries made into any forms.

(b) *Web server responses*: The response of a web storefront to a customer may be very important in understanding the customer experience. For example, to determine frustration stemming from difficult searches, it is important to know about both the number of searches performed and also the sizes of returned answers.

(c) *Enterprise data*: A broad variety of stored information may be useful to the personalization. At a minimum this will include accessing information resulting from bulk statistical analyses and information on inventory and availability times. If the customer has been identified then customer profiles and recent customer histories can also be incorporated into the personalization process.

There is also a higher, semantically rich kind of information that can be helpful as input for the decision engine. This will include information, for example, about the category of page being accessed by the customer (e.g., search request, search answer, catalog entry, shopping cart), or the intent of a page (e.g., that it includes a promotion or indicates that a certain catalog item is out of stock).

Indeed, an important part of installing a DFP system, or any web personalization system using on-line decision support, will involve creating or determining a model of the web site, that incorporates relevant models of customers, the intent of their activities on the site, the business value of those activities, indicators that a customer may abandon a transaction, etc. The example of Section 3 provides a starting point for such a model, but a variety of other factors may be brought into play. This model will be clearly visible in the program driving the decision engine, and will help to guide the kinds of information that need to be passed from web server to decision engine.

There is a trade-off between attempting to automatically infer semantically rich information from the HTML passed between customer and web server vs. manually incorporating that information into the web page generation so that it can be obtained easily by the decision engine. Attempting to infer this information automatically typically involves parsing the HTML; it will involve the development of special-purpose code and be computationally expensive. Further, its success will depend on how direct and uniform the relationship is between the actual HTML content of the web pages and their intent. On the other hand, incorporating code into the web page generation that captures the semantically rich information puts an additional burden on the web site developer, both at creation time and during maintenance. A site such as Amazon or Yahoo could have thousands of pages, some static, others generated dynamically via server-side scripting languages such as ASP/JSP, or CGI-scripts/servlets. It will be a huge effort to modify all executable scripts to add the MIHU functionality.

Sophisticated web authoring environments such as Microsoft's FrontPage or Allaire's ColdFusion Studio provide hooks so that web site authors can easily incorporate semantically rich information into the HTML generated by their code. Thus it would be straightforward for site developers to extract high-level semantic information to be passed to the Vortex engine. However, if the site has not been built using such tools, we expect that early adopters of our personalization technology will opt for parsing the HTML, and will use only some of the information actually available.

4.2 Acquiring the Information

In this subsection, we examine five techniques of gathering session information required for the decision engine, and discuss their pros and cons. A summary is presented in Figure 6. It is expected that a combination of these techniques will be required in a DFP toolkit, if it is to be deployed to support personalization for a broad variety of web sites. After presenting the techniques we make some general remarks.

Content generation scripts send high-level semantics to decision engine. Assuming that all pages that need to be tracked are generated via executable scripts/programs (which is a reasonable assumption to make for large sites), an obvious approach to obtaining meaningful semantic information would be to create or modify these scripts to gather/create the desired information, and then pass it on to the decision engine. The primary advantage of this approach is that the people developing the web pages will have the best idea of the intended semantics of the pages, and thus what the decision engine should receive. For this reason, we expect this to be the approach of choice when creating new web sites. Further advantages are that the actual HTTP requests and responses do not need to be transformed/parsed, and HTTPS connections can be handled. The primary disadvantage concerns legacy sites, where modifying all the existing scripts to generate the high level semantic data would be quite expensive. Another disadvantage is that maintenance of the site would become more cumbersome.

So how can the DFP approach be used in large legacy web sites? In such cases, the only solution might be to try and extract meaningful information from the raw HTTP requests/responses. There are various ways to do so, some of which we discuss below.

Strategy	Converting legacy sites easy	HTTPS handled?	HTTP response transformation required	HTTP response parsing required
Content generation scripts send high-level semantics to engine	No	Yes	No	No
Content generation scripts send raw HTML to engine	Yes	Yes	No	Yes
Wrapper scripts	Yes	Yes	Maybe	Yes
Proxies	Yes	No	No	Yes
Web Server Extensions	Yes	Yes	No	No

Figure 6: Comparison of Web Interaction Monitoring Strategies

Content generation scripts send raw HTML to decision engine. This is a variation of the approach mentioned earlier, however, in this case, only the raw HTTP requests/responses are forwarded by the scripts. This can be done by injecting the same (small) block of code into the scripts that generate each and every page of the web site. The advantage is that converting a legacy site to this approach is straight-forward (assuming that it was implemented with server-side scripting language such as JSP or ASP), since the only function the extra piece of code performs is to forward appropriate data (HTTP request and/or response) to the decision engine. However, since the injected code will be uniform and generic, it will not be able to extract high-level semantic information from each page. This means that detailed knowledge of what information to extract for specific (categories of) pages, and how to extract it, needs to be built, either in the decision engine or in some other process. Depending on the level of information to be extracted, this can cause maintenance problems anytime the structure of the corresponding pages change. Moreover, this approach is sensitive to the language and/or platform that the web site is implemented in, e.g., if the CGI scripts are based on C++ then it may be hard to know where to inject the code block, making the approach infeasible.

Wrapper scripts. The idea here is that the web server can be configured so that all web requests and responses (that need to be tracked) are filtered through executable scripts that perform the task of extracting the relevant information and contacting the Vortex server to determine the appropriate response. Note that these wrapper scripts could reside on the web server(s) that are supporting page requests, or could reside on separate machines. As opposed to the previous approaches, the advantage in this scenario is that the actual content generation is not affected — this method is simply layered on top. Moreover, HTTPS connections can be handled, since the wrapper script gets the customer request after it is decrypted by the server, and parses the response before it is encrypted and sent to the client. A disadvantage is that HTML pages being served would need to be transformed, since the links/forms/frames in existing pages now need to go through the wrapper, whereas other objects (e.g., pre-loaded images via Javascript) need to be accessed directly. However, it can be hard to automatically transform all underlying pages, especially if a lot of destination URL computation is done inside client-side script code, which would require the wrapper to parse the corresponding scripting language. Also, HTML pages input to the wrapper would need to be parsed and translated into higher level semantic information, either by the wrapper or the decision engine. Finally, session tracking information may be lost under this approach, if the web server is using a cookie-based scheme that tracks sessions for some but not all of the web site pages. In that case, replacing URLs so that they access the wrapper scripts may disrupt the web site's scheme for putting cookies at the customer site. To remedy this, some re-writing of the web site scripts would be required. However, this problem can be eliminated if the web server allows URL re-direction based on customizable rules. In that case, the customer could see the same URL and no HTML re-writing would be required, hence session tracking would not be a problem.

Proxies. A proxy can be inserted between a company's web site and the end user. The proxy is responsible for tracking user requests, extracting the site responses, and contacting the decision server to determine the appropriate intervention strategy. An advantage is that HTML page transformation is not required. However, there are several disadvantages to this approach. Firstly, if SSL tunneling is being used, then the proxy will need to serve as the receiving end of the tunnel, and will need to perform encrypting/decrypting of the web traffic. Moreover, it would also need to extract higher level semantic information from the HTML. Lastly, the use of one or more proxies may have impact on scalability, because the proxy servers can become a bottleneck. It will be important to have enough proxies to cover the anticipated load on the web site.

Web Server Extensions. Most popular web servers (Apache, Netscape Enterprise, Microsoft IIS) have an API (Apache modules, Netscape's NSAPI, Microsoft's ISAPI) that can be used to extend the functionality provided by the server. In particular, these can be used to attach monitoring hooks into the web server itself, thus gaining low-level access to all web interactions. The advantage is that no transformation of HTML response being generated is required, and secure connections can be handled. The disadvantages of needing to extract higher-level semantic information from HTML responses still applies. Moreover, writing server extensions is tricky (since they should not impact reliability or scalability) and server specific.

We conclude this subsection with some general remarks about these techniques and our experience with two of them.

We first consider session tracking. Three techniques are commonly used for tracking a session in web sites: encoding the session ID into the URLs sent and requested by the customer, placing cookies on the customer machine, and placing the session ID into a hidden form field. (The latter technique requires that all pages transmitted to the user are generated via form submissions.) In order for the decision

engine to know the session of a page request, the session ID must be passed to the engine along with other page information. The session ID can be sent explicitly, or it can be sent as it occurs in the HTML of the requested page, and the encoding scheme used by the web site can be used to extract the session ID.

We now turn to the issue of scalability. In particular, how do the above techniques work when a web site is supported by a web server farm rather than a single web server? There are two main issues. First, in the case of a web server farm there may also need to be a farm of decision engines. Because the log of a given customer session will generally be maintained in the main memory of a single decision engine, it will be important that all decisions about that session be made by the same decision engine, even if different web servers are being used to serve the pages. This can be accomplished by encoding the decision engine ID inside the session ID. A load-balancing strategy can be implemented to distribute customer sessions across the decision engines. Furthermore, in applications such as MIHU, if all of the decision engines reach saturation then the system can decide for some customers that they will not receive any MIHU decisions. This permits a graceful degradation of service in the face of unexpectedly high load.

The second scalability issue concerns how the added expense of transmitting information from web server to decision engine will impact performance. In all cases except for proxies, the processing involved in transmitting to the decision engine can be performed on the web server. Thus, each server will be more loaded, but no architectural problems arise. In the case of proxies (and wrapper scripts if they are implemented on separate machines) there is a possibility of the proxy becoming a bottleneck.

We have built two versions of the MIHU prototype at Bell Labs, that explored some of the issues discussed above. In the first case, we modified the content generation CGI-scripts used by the web site. In the second case, we wrote a wrapper servlet. Here, actual request URLs were passed to the wrapper servlet via its `PATH_INFO` environment variable. The servlet then performed the original request and parsed the HTML response generated to extract the relevant information. Before shipping the response to the customer, the servlet also modified the links/forms/frames in the page to go through the servlet, and inserted a `BASE` tag that pointed to the original URL so that any relative accesses (e.g., preloaded images inside Javascript) would work. None of the actual pages stored at the web site needed to be modified.

Importantly, with any of the above monitoring methods presented, the method can be phased into the site – e.g., initially the tracking can focus only on part of the site, and only on part of the relevant data.

4.3 Impacting the Web Site

So far, we have focused on tracking the user experience and using the on-line decision engine to recommend how the customer experience should be impacted. We now consider the kinds of recommendations that can be made, and how they can be acted upon.

A natural kind of recommendation is the placement of an icon or image on a page transmitted to the customer. The image could either correspond to a promotion, or could even offer the customer live agent help (for web sites with call centers). If the customer clicks on the image, the web server could assign a live agent to interact with the customer. These techniques are common in existing personalization approaches, and easily supported in our approach.

It is also possible for the decision engine to generate entire pages or frames. For example, at Bell labs we have used the Vortex engine to choose from a number of parameterized page templates, and then choose values for the parameters. In this way, simple automated conversations with the customer can be performed. Extending this technique to support more complete, highly personalized generation of web sites is an important direction, and will require the creation of a rich development environment, and perhaps another layer of abstraction on top of the current Vortex language.

5. RELATED WORK

This section compares the DFP approach with related work, including other e-commerce personalization solutions, other decision specification paradigms, and finally a system that provides some aspects of the MIHU functionality.

Existing e-commerce personalization tools based on on-line decision support use rules languages that are quite limited in expressive power. For example, Manna [15] provides an event-condition-action rules language, where the actions result in side effects outside of the rules system. There is no chaining of rules, which limits the expressive power. For example, it is not feasible in these systems to use a cluster of rules to compute a business opportunity score, another cluster of rules to compute a customer frustration score, and a final cluster of rules that combines the two scores and other information to select an appropriate action. The scripting language of Blaze [2] provides flowchart constructs, where a node in a flowchart may contain a set of rules. Inside such flowchart nodes there is no chaining of the rules. And so any rule chaining that needs to be expressed is explicit in the use of flowchart constructs between the rules nodes. As a result, users must explicitly specify essentially all flow of control for the decision-making process. For complex decisions this is too cumbersome for business analysts and managers.

What about using some other, existing decision specification system to support e-commerce personalization? Decision trees are too confining in their logic. Logic programming [14], including variants that incorporate negation (e.g., [21, 10]), while Turing complete, is both too expressive (because it is hard to develop reports that easily explain how decisions are reached) and too confining (because it forces a single semantics for combining rules which makes it hard to directly express both formal and heuristic styles of reasoning). Expert systems (e.g., OPS5 [3]) are also too expressive, because it is difficult to explain how decisions are reached, and difficult to predict the overall effect of modifying a rule. One candidate that met several of the user requirements is the RAISE system [9]; however, the adherence to a logic-programming style of rules did not support directly expressing both formal and heuristic styles of reasoning.

Personalization based on off-line decision support (e.g., Epiphany [8], Net Perceptions [17]) performs periodic bulk data analysis using data mining and statistical techniques to infer correspondences between, e.g., customer types, products already selected, and products that would be appropriate for targeted promotion. As with other on-line approaches, DFP is complimentary to the off-line approach, and decisions made using DFP can access the results of off-line analysis.

Another tool that can be used for on-line decision support is described in [16], which presents a formalism for scoring the quality of data assembled from multiple sources. The formalism uses vectors giving scores to various criteria, along with operators to combine vectors; these combinations correspond to combinations of scientific data performed in the underlying workflow. The Decision Flow paradigm can express this formalism, because it supports direct manipulation of record types, and can include the operators for combining vectors as named combining policies.

We contrast our approach with another popular approach for customizing web-sites, that is based on guiding the customer through a series of questions that are used to identify customer preferences and their relative weights (e.g., Personalogic [19]). More generally, [1] presents a framework for specifying and combining preferences, that provides high flexibility and satisfies mathematical properties such as closure under certain operations. The Decision Flow paradigm can simulate this preference model by using record structures and specialized combining policies.

The DFP approach consists of using an on-line decision engine that is separate from the web server. An alternative would be to use some form of server-side scripting language (e.g., ASP, JSP, or PHP), that could hold the business logic for on-line decision making. The fundamental problem here is that these scripting languages do not satisfy several of the requirements on the decision specification language, such as the explicit presence of rules, the ability to combine information in different ways, or the correspondence between programs and reports.

Finally, the MIHU system presented here differs from previous systems for presenting live CSR assistance to customers. For example, iContact [13] uses a simple rules mechanism to identify customer sessions on a web store-front that are "good" candidates for live CSR assistance. However, with iContact the CSRs are given a listing of these candidates, and the CSRs make the final decision about whether or not to offer live intervention. With the MIHU system the entire decision can be automated (taking into account not only the business opportunity afforded by the customer, but also the current availability of CSRs to help realize this opportunity). This permits treatment of customers which is more uniform than with the iContact approach. The MIHU system also permits more careful selection of a CSR whose skill matches the expected needs of a given web-based customer. Finally, it can support more sophistication in the use of "blended" CSRs, who spend some time with web-hosted customers and other time with traditional telephone-based customers.

6. DISCUSSION AND FUTURE WORK

This paper presents the DFP framework for personalizing web sites that is based on the use of an expressive on-line decision engine. Key requirements on the language used by the decision engine were identified, and the Vortex language with Decision Flows was introduced to satisfy these requirements. Various approaches to implementing our approach on new or existing web sites were explored, and our experiences with implementing some of them discussed. Finally, the paper described the May-I-Help-You prototype system, which applies the technology to decrease the number of e-commerce transactions that are abandoned. We close by mentioning several directions for future work.

VorteXML. The current Vortex language and engine is geared primarily towards relational data. We are working to extend Vortex so that it can also work with XML-based data, thus allowing a uniform paradigm for converting the raw data into higher-level semantic information and for analyzing that higher-level information (see [5]). There are various ways in which a web site can pass data to the decision engine. For example, the web server might do some initial processing and cleaning to transform the HTML to XML (e.g., using XPath [6]), and then pass the XML to the VorteXML engine. The ability for Vortex to specify various heuristics would be useful in converting that XML into higher level semantics, especially for web sites that represent information in non-uniform ways.

Another approach is to *annotate* the HTML content produced by a web site, by adding custom tags that are ignored by the client browsers, but which the decision engine can scan to extract the desired information. This would make the task of extraction easier, since the decision engine now only needs to look at a subset of the raw input.

There is also a move towards separating content from presentation on web sites, i.e., for each customer request, a web site would retrieve the actual content as an XML document, and then apply an XSL stylesheet to transform it into an HTML document before shipping it to the client. In such a case, the web site could simply forward the XML content to the decision engine, which would remove the task of transforming the HTML into XML.

Distributed Rules Processing. Section 4 described how DFP can be scaled, in an architectural sense, to environments with web server farms. Another challenge concerns scaling to large web sites, as found in the B2B sites of large corporations. These typically span multiple sub-organizations and span multiple geographic locations. For example, in Lucent a customer might enter the Lucent home page that is supported by web servers in New Jersey, but then access product information about the latest IP telephony switches via web servers in Illinois. Furthermore, while some decision policies might be applicable to all customers, others might be relevant only to certain products. This means that the rule sets used in connection with different locations may be overlapping but different. The challenge is to support the development of such overlapping rule sets, have appropriate rules apply to the pages being examined, and pass relevant data between geographic locations as the customer's session moves between those locations.

Reliability. We are currently experimenting with the use of fault-tolerant CORBA to provide reliability for the Vortex engine, and thus for the DFP approach. Another way to achieve reliability, and scalability for that matter, would be to implement the Vortex language as part of an application server platform (e.g., based on EJB).

Automated learning. Following the lead of companies such as Manna [15], it will be important to incorporate automated learning into any personalization technology. Because Vortex provides rules-constructs that are richer than many business rules systems, it will be more difficult to develop learning technology for Vortex. On the other hand, the structure of the rule sets and the availability of meaningful reports should provide important handles to the problem.

Acknowledgments

The authors thank Greg Anderson of Avaya Communication for suggesting "May I Help You" as a useful test case and illustration of the Vortex Decision Flow technology.

7. REFERENCES

[1] R. Agrawal and E. Wimmers. A framework for expressing and combining preferences. In *Proc. ACM SIGMOD Symp. on the Management of Data*, pages 297–306, 2000.

[2] Blaze Software Home Page, 2000. `http://www.blazesoft.com`.

[3] L. Brownston, R. Farrell, E. Kant, and N. Martin. *Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming.* Addison-Wesley, Reading Massachusetts, 1985.

[4] R. Cattell. *The Object Database Standard: ODMG-93.* Morgan Kaufmann Publishers, San Mateo, California, 1993.

[5] V. Christophides, R. Hull, A. Kumar, and J. Siméon. Workflow mediation using VorteXML. *IEEE Data Engineering Bulletin*, 24(1), March 2001.

[6] J. Clark and S. DeRose. XML Path Language (XPath). Technical report, World Wide Web Consortium, 1999. W3C Recommendation 16 November 1999.

[7] Datamonitor, Inc. The U.S. market for internet-based customer service, 2000.

[8] Epiphany Home Page, 2000. `http://www.epiphany.com`.

[9] B. N. Grosof, D. W. Levine, H. Y. Chan, C. J. Parris, and J. S. Auerbach. Reusable architecture for embedding rule-based intelligence in information agent. In *Proceedings of the Workshop on Intelligent Information Agents*, December 1995. Held in conjunction with the ACM Conference on Information and Knowledge Management (CIKM-95).

[10] G. N. Grosof. Compiling prioritized default rules into ordinary logic programs. Technical Report RC 21472, IBM Yorktown, May 1999.

[11] R. Hull, F. Llirbat, B. Kumar, G. Zhou, G. Dong, and J. Su. Optimization techniques for data-intensive decision flows. In *Proc. IEEE Intl. Conf. on Data Engineering*, pages 281–292, 2000.

[12] R. Hull, F. Llirbat, E. Simon, J. Su, G. Dong, B. Kumar, and G. Zhou. Declarative workflows that support easy modification and dynamic browsing. In *Proc. of Intl. Joint Conf. on Work Activities Coordination and Collaboration (WACC)*, pages 69–78, February 1999.

[13] iContact Home Page, 2000. `http://www.icontact.com`.

[14] J. W. Lloyd. *Foundations of Logic Programming (Second Edition).* Springer-Verlag, Berlin, 1987.

[15] Manna Home Page, 2000. `http://www.mannainc.com`.

[16] F. Naumann, U. Leser, and J. Freytag. Quality-driven integration of heterogeneous information systems. In *Proc. of Intl. Conf. on Very Large Data Bases*, pages 447–458, 1999.

[17] Net Perceptions Home Page, 2000. `http://www.netperceptions.com`.

[18] D. Peppers and M. Rogers. *The One to One Future.* Doubleday, New York, 1993.

[19] Personalogic Home Page, 2000. `http://www.personalogic.com`.

[20] P. B. Seybold and R. T. Marshak. *customers.com.* Random House, New York, 1998.

[21] A. Van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *J. ACM*, 38(3):620–650, 1991.

PORTAL ACM DIGITAL LIBRARY

> home > about > feedback > login

US Patent & Trademark Office

Try the *new* Portal design

Give us your opinion after using it.

Citation

International World Wide Web Conference >archive
Proceedings of the tenth international conference on World Wide Web >toc
2001 , Hong Kong, Hong Kong

Personalizing E-commerce applications with on-line heuristic decision making

Authors
Vinod Anupam
Richard Hull
Bharat Kumar

Sponsors
SIGWEB : ACM Special Interest Group on Hypertext, Hypermedia, and Web
SIGLINK : Hypertext, Hypermedia, and Web
IW3C2 : International World Wide Web Conference Committee

Publisher
ACM Press New York, NY, USA

Pages: 296 - 307 Series-Proceeding-Article
Year of Publication: 2001
ISBN:1-58113-348-0

doi> http://doi.acm.org/10.1145/371920.372073 (Use this link to Bookmark this page)

> full text > references > index terms > peer to peer

> Discuss > Similar > Review this Article Save to Binder

> BibTex Format

↑ **FULL TEXT:** Access Rules

pdf 261 KB

↑ **REFERENCES**

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

1 Rakesh Agrawal , Edward L. Wimmers, A framework for expressing and combining preferences,

Proceedings of the 2000 ACM SIGMOD international conference on Management of data, p.297-306, May 15-18, 2000, Dallas, Texas, United States

2 Blaze Software Home Page, 2000. http://www.blazesoft.com.

3 Lee Brownston , Robert Farrell , Elaine Kant , Nancy Martin, Programming expert systems in OPS5: an introduction to rule-based programming, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1985

4 R. Cattell. The Object Database Standard: ODMG-93. Morgan Kaufmann Publishers, San Mateo, California, 1993.

5 V. Christophides, R. Hull, A. Kumar, and J. Simeon. Work ow mediation using VorteXML. IEEE Data Engineering Bulletin, 24(1), March 2001.

6 J. Clark and S. DeRose. XML Path Language (XPath). Technical report, World Wide Web Consortium, 1999. W3C Recommendation 16 November 1999.

7 Datamonitor, Inc. The U.S. market for internet-based customer service, 2000.

8 Epiphany Home Page, 2000. http://www.epiphany.com.

9 B. N. Grosof, D. W. Levine, H. Y. Chan, C. J. Parris, and J. S. Auerbach. Reusable architecture for embedding rule-based intelligence in information agent. In Proceedings of the Workshop on Intelligent Information Agents, December 1995. Held in conjunction with the ACM Conference on Information and Knowledge Management (CIKM-95).

10 G. N. Grosof. Compiling prioritized default rules into ordinary logic programs. Technical Report RC 21472, IBM Yorktown, May 1999.

11 R. Hull, F. Llirbat, B. Kumar, G. Zhou, G. Dong, and J. Su. Optimization techniques for data-intensive decision ows. In Proc. IEEE Intl. Conf. on Data Engineering, pages 281-292, 2000.

12 Richard Hull , Francois Llirbat , Eric Siman , Jianwen Su , Guozhu Dong , Bharat Kumar , Gang Zhou, Declarative workflows that support easy modification and dynamic browsing, Proceedings of the international joint conference on Work activities coordination and collaboration, p.69-78, February 22-25, 1999, San Francisco, California, United States

13 iContact Home Page, 2000. http://www.icontact.com.

14 J. W. Lloyd, Foundations of logic programming; (2nd extended ed.), Springer-Verlag New York, Inc., New York, NY, 1987

15 Manna Home Page, 2000. http://www.mannainc.com.

16 F. Naumann, U. Leser, and J. Freytag. Quality-driven integration of heterogeneous information systems. In Proc. of Intl. Conf. on Very Large Data Bases, pages 447-458, 1999.

17 Net Perceptions Home Page, 2000. http://www.netperceptions.com.

18 D. Peppers and M. Rogers. The One to One Future. Doubleday, NewYork, 1993.

19 Personalogic Home Page, 2000. http://www.personalogic.com.

20 P. B. Seybold and R. T. Marshak. customers.com. Random House, New York, 1998.

21 Allen Van Gelder , Kenneth A. Ross , John S. Schlipf, The well-founded semantics for general logic programs, Journal of the ACM (JACM), v.38 n.3, p.619-649, July 1991

↑ INDEX TERMS

Primary Classification:

I. Computing Methodologies

↳ **I.2** ARTIFICIAL INTELLIGENCE

↳ **I.2.8** Problem Solving, Control Methods, and Search

↳ **Subjects:** Heuristic methods

Additional Classification:

H. Information Systems

↳ **H.3** INFORMATION STORAGE AND RETRIEVAL

↳ **H.3.5** On-line Information Services

↳ **Subjects:** Web-based services

↳ **H.4** INFORMATION SYSTEMS APPLICATIONS

↳ **H.4.2** Types of Systems

↳ **Subjects:** Decision support (e.g., MIS)

↳ **H.5** INFORMATION INTERFACES AND PRESENTATION (I.7)

↳ **H.5.3** Group and Organization Interfaces

↳ **Subjects:** Web-based interaction

K. Computing Milieux

↳ **K.4** COMPUTERS AND SOCIETY

General Terms:

Algorithms, Design, Management, Measurement, Performance, Theory

Keywords:

B2C E-commerce, personalization, pro-active intervention, vortex rules system

↑ Peer to Peer - Readers of this Article have also read:

- Augmenting shared personal calendars
  Proceedings of the 15th annual ACM symposium on User interface software and technology
  Joe Tullio , Jeremy Goecks , Elizabeth D. Mynatt , David H. Nguyen

- M^4: a metamodel for data preprocessing
  Proceedings of the 4th ACM international workshop on Data warehousing and OLAP
  Anca Vaduva , Jörg-Uwe Kietz , Regina Zücker

- Polymer simulation on the hypercube

Proceedings of the third conference on Hypercube concurrent computers and applications
H-Q. Ding

- Data structures for quadtree approximation and compression
  Communications of the ACM 28, 9
  Hanan Samet

- A hierarchical single-key-lock access control using the Chinese remainder theorem
  Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing
  Kim S. Lee , Huizhu Lu , D. D. Fisher

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

PORTAL
THE ACM DIGITAL LIBRARY

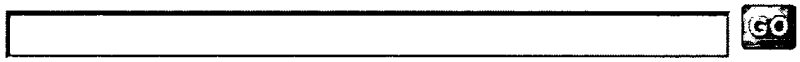
> home > about > feedback > login
US Patent & Trademark Office

Try the *new* Portal design
Give us your opinion after using it.

Search Results

Search Results for: **[ecommerce<AND>((rule system))]**
Found **2** of **129,763 searched.**

Search within Results



> Advanced Search

> Search Help/Tips

Sort by: **Title** **Publication** **Publication Date** **Score** Binder

Results 1 - 2 of 2 **short listing**

1 Personalizing E-commerce applications with on-line heuristic decision making 85%
Vinod Anupam , Richard Hull , Bharat Kumar
Proceedings of the tenth international conference on World Wide Web April 2001

2 Pushing reactive services to XML repositories using active rules 77%
Angela Bonifati , Stefano Ceri , Stefano Paraboschi
Proceedings of the tenth international conference on World Wide Web April 2001

Results 1 - 2 of 2 **short listing**

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

Pushing Reactive Services to XML Repositories using Active Rules

Angela Bonifati
Politecnico di Milano
Piazza Leonardo Da Vinci 32
I-20133 Milano, Italy
bonifati@elet.polimi.it

Stefano Ceri
Politecnico di Milano
Piazza Leonardo Da Vinci 32
I-20133 Milano, Italy
ceri@elet.polimi.it

Stefano Paraboschi
Politecnico di Milano
Piazza Leonardo Da Vinci 32
I-20133 Milano, Italy
parabosc@elet.polimi.it

ABSTRACT

Push technology, i.e., the ability of sending relevant information to clients in reaction to new events, is a fundamental aspect of modern information systems; XML is rapidly emerging as the widely adopted standard for information exchange and representation and hence, several XML-based protocols have been defined and are the object of investigation at W3C and throughout commercial organizations. In this paper, we propose the new concept of active XML rules for "pushing" reactive services to XML-enabled repositories. Rules operate on XML documents and deliver information to interested remote users in reaction to update events occurring at the repository site.

The proposed mechanism assumes the availability of XML repositories supporting a standard XML query language, such as XQuery that is being developed by the W3C; for the implementation of the reactive components, it capitalizes on the use of standard DOM events and of the SOAP interchange standard to enable the remote installation of active rules. A simple protocol is proposed for subscribing and unsubscribing remote rules.

Categories and Subject Descriptors

H.2 [**Database Management**]: Languages; I.7 [**Document and Text Processing**]: Document Management

General Terms

Languages

Keywords

Push technology, Active Rules, XML, SOAP, Document management, Query languages for XML

Copyright is held by the author/owner.
WWW10, May 1-5, 2001, Hong Kong
ACM 1-58113-348-0/01/0005.

1. INTRODUCTION

One important aspect of Internet-based information systems is the ability of pushing information to clients, by matching new event occurrences with predefined user's interests. Such ability is embedded within many WEB development products [23, 8] and applications [49, 3], which support one-to-one information delivery in response to users' current and past interactions. Active rules and database triggers are an important ingredient for supporting this reactive technology [1]. All of the above proposals, however, make use of mechanisms which operate locally, on top of given data sources which are controlled by the organizations delivering the information to users. So far, the possibility of distributing the "pushing logic", and installing it at remote servers, has not been considered.

In this paper, we argue that such a possibility is becoming very concrete with the advent of new technological standards, such as XML [7], and XML query languages [33, 10, 14, 21, 39, 17, 15], and with the parallel development of XML-based repository technology [43, 38, 19]. Furthermore, the Internet and Web communities are repeatedly proposing the use of XML in network protocols and distributed applications - XML-RPC [48], SOAP [41], XMI [46], ebXML [25], ICE [31], IOTP [32] and XML Protocol [47] are only a few examples.

Our proposed approach falls under the generic framework of e-services; such a paradigm denotes a class of Internet computations and systems which fulfill a given objective with some degree of autonomy, for instance because they search within the Internet the best matchings of given client requests, or are capable of simple forms of negotiations. Along these lines, we propose a class of Internet services that behave, in a remote system, by means of active rules; these rules monitor the events occurring at the remote systems and notify interested information consumers. Each rule acts like an independent e-service; a B2B protocol regulates the remote installation of rules at the server, which is proposed by a rule broker and accepted by the remote repository; this negotiation follows a simple installation contract.

The rules that we propose in this paper are not currently supported by XML repositories; however, the standard bodies, and particularly the W3C, are making the appropriate steps in order to make the implementation of such rules rather simple. In particular, active XML rules capitalize on the existence of events in DOM (since the Level 2 Specification [22]) and of a standard XML query language, named XQuery, which has recently been proposed by W3C XML

Query Working Group. Our proposal is independent from the choice of a particular XML query language, and is currently based on the XQuery Working Draft [15].

We have already discussed, in [6], the issues that arise in the development of active rules for XML. We have presented two specific instantiations of active rules, relative to Lorel and XSL used as query languages; and we have studied the issues of rule conflicts and of their properties, such as termination, confluence, and edit-script independence. It is worth noticing that many of the problems discussed in [6] relative to a generic XML active rule set are much simplified in the environment proposed in this paper, because the active rules that we use have only the ability of notifying remote users, and therefore cannot trigger each other. Thus, termination is guaranteed; conflict resolution policies may determine different orders of notifications to subscribers.

The submission of a rule for its execution by the server permits to locate tasks near to the data, which is innovative for the Web context; this is similar to what happens in distributed databases with stored procedures, that locate applicative code within the server maintaining the data [11]. This integration guarantees the fastest possible notification to subscribers, who come to know of events as they occur; it also improves the global systems' efficiency, because services are executed right where the information resides, without requiring expensive data replication.

For the negotiation protocol, and specifically for the interchange between the rule broker and the XML-based repository, we use SOAP protocols and envelopes. The use of SOAP as generic e-service invocation mechanism makes our solution flexible and portable. SOAP was chosen among the several available XML-based protocols, due to its increasing popularity as a lightweight protocol for exchange of information in a decentralized environment. The XML Protocol Working Group at W3C is addressing the specification of requirements of XML Protocol, which condenses and extends the experience of previously defined lightweight protocols, including SOAP. Such a protocol, once deployed, can be easily adopted in our framework.

This paper is organized as follows. After an overview of related work in Section 2, Section 3 briefly presents the syntax and semantics of XML active rules. Section 4 describes the application scenario for rule brokering. Section 5 describes the B2B protocol for submitting a rule to the XML Server. Section 6 describes the steps that are needed in order to implement a reactive engine on top of an XML Server. Finally, Section 7 draws the conclusions.

2. RELATED WORK

Event-based computation has been studied in several diversified communities, spanning from software engineering and networking to databases. Within these communities, many event-based distributed architectures have been defined implementing the mechanisms of remote event subscription, filtering and management. Among these systems, we cite OMG Event and Notification Service [36, 37], Yeast [35], Ready [29] and Smarts [42]. OMG Event Service and Notification supports asynchronous exchange of events among clients, which can play the roles of event consumers or suppliers and use event channels to communicate. The publishing mechanism of events is based on the pull/push model and a filter facility is provided. In Yeast, event patterns are descriptors, actions are sequences of commands and remote invocations; the underlying communication layer is based on a traditional client/server paradigm. Yeast is extended by Ready, which introduces a specification language for matching of events and quality of service directions. Smarts deploys a distributed system architecture for the purpose of detecting and handling system problems. Event streams occurring in a network are elaborated in real-time and ad-hoc policies are adopted to solve problems. We have taken into account the experience of such event-based computation systems, particularly for what concerns our proposal of a rule subscription mechanism.

Within the database community, a lot of attention has been paid to reactive mechanisms broadened to a distributed environment [44]. Beyond the traditional applications of centralized active database systems, such as support of integrity constraints, materialized views, and derived data, reactive mechanisms can be used to implement services required for network management, e.g., mail services and firewalls. Prominent works concern the maintenance of materialized views in data warehousing systems [45, 50], or the constraint maintenance in a distributed environment [12] by means of distributed triggers. However, the application of active rules to the development of reactive push services has not yet been described in the literature.

The unbundling trend in the database field advocates the modularization of monolithic databases into smaller and autonomous services to promote more flexibility and functionality. According to this trend, an event service and a rule service can be enucleated from conventional databases and offered to the nodes of a network, guaranteeing portability and heterogeneity [13, 28, 34]. In our approach, unbundling is helpful to identify the rule components, but it is not strictly required, as we assume that XML rules will be part of the XML repository.

We discussed the application of XML active rules to implement suitable e-services in [6]; here, active rules implement business applications (such as alerters, personalizers and classifiers) as well as document maintenance. An event detector based on DOM [22] is responsible for capturing the data mutating events on the document and an XML query (expressed in Lorel [33] or in XSL [17]) implements the conditions and actions of the rules. In the present work, XQuery [15] has been chosen to encode XML active rules.

Other works deal with the definition of triggers for XML data, more or less concerned with e-commerce applications. A novel view specification language, equipped with active capabilities has been defined in [1]. The actors involved in an electronic commerce application might need different views of the repository data, and these are encoded through a set of activity specifications, methods and triggers. Enhanced mechanisms for notification, access control and logging/tracing of user activities are provided. Here active rules are application-specific and use a set of proprietary method calls, defined within the views.

Database reactive technology can be considered as complementary to agent technology in implementing event-based computation. Among agent-based systems, Jedi [20] is relevant because it is based on reactive objects, i.e. autonomous computational units performing application specific tasks. Each reactive object has its own thread of control, and interacts with other reactive objects producing and consuming events. They are based on mobile pieces of software (i.e., agents) and use a subscription facility to declare their

interest to events. Compared to agents, rules can be more easily implemented, since they are plugged into the XML repository without any need of mobility. Rules are less sophisticated than agents and offer less flexibility, but in our application context (i.e. push technology), they can be considered both light-weighted and enough powerful.

A language-specific and proprietary architecture to provide sophisticated e-services is the purpose of many high-tech industries research. Among the many products available on the market, e-speak [24] introduces several innovative aspects. The e-speak architecture leverages on a message passing mechanism between the e-speak logical machine (or core) and the resources, which provide the services. The communication is obtained through a specific session layer security protocol, and a mailbox metaphor is used to describe the interactions between the clients and the core. E-speak does not support reactive mechanisms.

The use of rules to perform e-services may produce some scalability issues. In fact, as the application of triggers moves from databases towards the Internet, the number of potentially expensive triggers becomes larger and efficiency becomes increasingly desirable. Recent works [30, 16, 2] discussed the theme of scalability applied to triggers and to continuous query systems. Their common idea is to share common computations among large sets of similar triggers or queries. These techniques could be adapted to the delivery of efficient and scalable XML rule engines. In our framework, rules do not interfere with each other, therefore scalability issues are less relevant than in traditional rule engines.

3. ACTIVE RULES FOR XML

The event-condition-action paradigm for active rules has demonstrated in the database context its flexibility and expressive power; each rule is characterized by the events that can "trigger" it; once a rule is triggered, the condition is "considered"; if the evaluation of the condition is successful, the rule action is executed. This model is proposed for XML active rules. A simple rule can be represented by the following XML fragment:

```
<event>insert(//cd)</event>
<condition> FOR $a IN //cd
            WHERE $a=$new AND
              $a/price < 20 AND
              contains($a/author,"Milli Vanilli")
</condition>
<action>
  <soap-header>
    <uri>/notification</uri>
    <host>131.175.16.105</host>
    <soap-action>notify</soap-action>
  </soap-header>
  <SOAP-ENV:Envelope
    xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
                                soap/envelope"
    SOAP-ENV:encodingStyle="http://
              schemas.xmlsoap.org/soap/encoding">
   <SOAP-ENV:Body>
    <m:Notify xmlns:m="http://
                      131.175.16.105/methods">
     <cdfound>
        $a//*
     </cdfound>
    </m:Notify>
   </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
</action>
```

The rule is triggered by an insertion of a <cd> element. The rule verifies if the sub-element <price> of the new element has a value less than 20 and if the element <author> contains the string "Milli Vanilli". If the condition is verified, the rule invokes the SOAP method Notify on the server 131.175.16.105, passing as parameter <cdfound> the inserted <cd> node and its content. We now give a more detailed description of each part of the rule.

3.1 Event

The *Event* part of the rule specifies the event responsible of rule triggering; it is enclosed into XML <event> tags. A *mutating* event is generated when the XML content is modified; we assume three types of mutating events: *insert*, *delete*, and *update*. The definition of a mutating event declaratively describes the nodes (elements or attributes) whose modifications need to be monitored; every time a monitored modification occurs, the corresponding event instance is generated and associated with the modified node. E.g., an event definition *insert(//house)* monitors the insertion of <house> elements in the repository; an event instance for the event is generated whenever a <house> element is introduced.

The Document Object Model (DOM) is an API defined by the W3C to access and manipulate XML information; the DOM consists of an object oriented model defined in IDL, that associates a set of methods with the nodes of an XML document. The component of the DOM interface most relevant to the event part of rules is the *Event Model*, introduced in the DOM Level 2 specification. The DOM Event Model assumes that the visualization and manipulation of XML information generates events on the nodes. For the realization of our services, we are interested into the *mutating events*, which are generated when a node of an XML document is modified (either because it is inserted, deleted, or its textual content is modified). Event detection is realized by *event listeners* associated with DOM nodes, which detect events occurring on the nodes to which they are associated or on their descendents. The detection of events on node descendents is based on a bi-directional propagation of the events, which requires that every event navigates downwards from the root of a document to the node instance on which the event occurred; when the event reaches its target, it can propagate back (*bubble* up) to the document root; events may be *captured* by event listeners; event listeners may choose to stop the propagation of events.

The availability of this sophisticated Event Model in DOM offers most of the services required for an implementation of a reactive mechanism for XML. The main problem that this event model presents is that event listeners are associated with node instances, whereas the event part of rules is defined in terms of a declarative specification of the schema element (e.g., a rule should be defined as triggered by an update on the price attribute of a car element). Thus, the implementation of events on top of the DOM Event Model requires the introduction of a conversion mechanism able to map each declarative event in an adequate set of associations of DOM event listeners with nodes.

3.2 Condition

The *Condition* part of the rule specifies the predicate that must be satisfied to execute the rule's action, expressed through a query which is interpreted as a truth value if it returns a nonempty answer. An important feature is the presence of a communication mechanism between the condition and the event part, so that the condition has a way to refer to the nodes on which the events occurred. This communication is based on predefined variables `new` and `old` that represent the nodes on which the events occurred with their current and past values, in a way similar to transition variables of database triggers.

An XML query for the condition can be expressed by using one of the available query languages for XML. The two most interesting alternatives are XPath and XQuery (in [6] we considered Lorel in place of XQuery). XPath is the language that permits the identification of nodes on which XSL templates must operate their transformation; XPath is a readable and intuitive language and has already gained an extensive support (e.g., the Xalan tool produced by the Apache Software Foundation).

The XML Query Algebra [27] is being developed by the W3C XML Query Working Group and is a compact procedural and strong-typed language for XML. Nonetheless, with respect to XPath, the XML Query Algebra is more a semantic and formal specification rather than an optimizable target language for XML.

XQuery [15] is the first step towards the definition of a standard XML query language, based on the experience of Quilt [14]. XQuery uses as one of its components XPath, and enriches it with services that permit the construction of an arbitrary XML structure as the result of a query. Therefore, XQuery is a good prototype of an expressive query language that is needed for the design of complex rules, used throughout this paper.

The implementation of the predefined variables `new` and `old` that represent the communication channel between the event and the condition depends on the choice of *row* or *set* semantics for the rules. The discussion on this aspect appears in [6], where we describe the two alternatives. Since the row semantics is easier to implement and to use and suited to the "pushing service" application, we assume a row-level semantics; thus, each event generates a rule consideration, in the context of which the variables `new` and `old` offer a reference to the node involved in the event. The implementation of these transition variables can take advantage of the `target` attribute of the `MutationEvent` interface and specifically of the `prevValue` and `newValue` attributes.

3.3 Action

The *Action* part of a rule specifies a SOAP method to invoke when the rule condition is evaluated true. We restrict the SOAP method to implement the call to a message delivery system, that will transfer information to specific recipients. In this way, the action part is much simpler than the general case, as discussed in [6]; in particular, there are no mechanisms for updating the content of the repository, or problems related to rule termination. In this proposal we assume that rules are not prioritized; therefore, if the same event may be serviced by several rules we cannot assume a rule execution ordering, and rule execution is not confluent; however, the addition of a simple prioritization mechanism could take place easily.

The implementation of the communication channel between condition and action is realized by permitting the reuse of condition variables in the action. The system will then replace at run-time each variable with the XML structure created by the query evaluation.

We assume that complex SOAP parameters can be passed to the method being invoked; these parameters are constructed in the condition and passed to the action, thus keeping the action very simple. It may be useful to define constraints limiting the recipients of the SOAP call appearing in the action of a rule to be authorized addresses, thereby introducing security requirements. It is possible to use mechanisms similar to those present in Java applets, for restricting the applet to the invocation of services available on the site from which the applet was downloaded. In general, the work done in the implementation of Java and in the CORBA middleware can offer many valuable suggestions to the design of a distributed execution system like the one we propose, particularly with respect to security aspects.

4. APPLICATION SCENARIO

The generic architecture of Reactive E-services Architectural Framework consists of three main actors (see Figure 1): Service Reseller, Service Supplier, and Rule Broker. The *Service Supplier* delivers goods and services described by an XML Server which internally supports an XML rule execution engine. Rules monitor events - such as the new availability of a service - and then notify the service resellers. The *Service Reseller* is, in turn, the recipient of messages concerning new services; they typically interact with clients which are interested in purchasing specific goods or services. The *Rule Broker* acts as an intermediary; it receives information about the services being searched over the Internet, and installs rules at the service supplier sites. For doing so, it needs to compose the rules in a suitable format and then to install them remotely.

There is no need of a standard protocol offered by the Rule Broker to the Service Reseller; simple WEB-based interfaces may be used in order to acquire information concerning the service being searched. Similarly, the Service Reseller receives a simple message informing of the relevant occurred events, without requiring any special purpose interface. However, the interface between the Rule Broker and the Service Supplier is a classical B2B interface, as the two systems need to establish a protocol which is both technical (yielding to the installation or removal of XML rules at the servers) and business-oriented (yielding to the establishment of mechanisms by means of which the rule broker sees its efforts being repaid). For these reasons, we propose a B2B interface between the Rule Broker and the Service Supplier, based upon four SOAP primitives (*Connect*, *Subscribe*, *Unsubscribe* and *Disconnect*) that are invoked by the Rule Broker and that are supported by the XML Server. We denominate this interface the Rule Submission Protocol and describe it in Section 5.1.

The XML Server needs to be augmented in order to support the protocol mentioned above and to execute XML rules; Section 6 is dedicated to describe the adaptations which are needed in order to upgrade the XML server technology.

The Rule Broker has a double role in the architecture: it is assigned the task of rule creation and it is responsible for the submission of the rules to all the XML Servers

that can contribute to the service. Rule creation requires that the Rule Broker knows how to write rules that satisfy a given applicative need and how to submit them to XML Servers using the Rule Submission Protocol. Rule submission requires that the Rule Broker knows which are the XML Servers that store the pieces of information that are relevant for the application, possibly through interaction with the Service Reseller. In summary, the Rule Broker is the *mediator* of the business transaction that is realized by the reactive services.

The Service Reseller may be directly the user of the notification server, or a mediator which presents the service to the final user with a user-friendly interface.

4.1 Example of Application

The architectural framework will be shown at work with a concrete example, in the field of real estate agent applications. An example of request can be: *a furnished four-bedroom (or more), two-bathroom (or more) Victorian house, which costs $1,500,000 or less, located in the Marina area in San Francisco.* A request is immediately issued on a list of house agency XML Servers, but no matching is found. Then, the reactive service is invoked; as a result, a set of rules are submitted to the XML Servers of several house agencies with the task of promptly reacting when a house, which satisfies the requirements, becomes available on the market. To achieve such result, a request is sent from the reseller to the rule broker, a contract is defined upon the rule and the proper authorization for the use and installation of the rule is marshalled, passing through the Rule Subscription Protocol commands.

The message exchange between the Rule Broker and the Reseller is a Rule Subscription Request. The communication and coordination between the Rule Broker and the Reseller can be managed as well through a web interface. After the negotiation, a response can be sent back to the reseller to confirm or reject the subscription. In the following, an example of Rule implementing the request is reported and a Rule-Generated Message delivered by the XML Server to the Reseller, is shown. The information that is sent out to the XML Server is encoded in SOAP and XML; it encloses the rule, the type of contract and the identification of the reseller. The event part is based on any update of relevant elements in the XML resources of the real estate agency servers. The condition part represents the query issued by the final user and is expressed in XQuery; the action part is the invocation of a SOAP method which takes care of alerting the reseller with the news about the houses appearing on the market.

```
POST /Soap/Rules/Subscribe HTTP/1.1
Host: www.expensivehousesincalifornia.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "/Subscribe"

<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
                                soap/envelope/"
   SOAP-ENV:encodingStyle="http://
          schemas.xmlsoap.org/soap/encoding/">
 <SOAP-ENV:Body>
  <m:Subscribe xmlns:m="http://
             schemas.xmlsoap.org/soap/rules/">
   <openConnection>343</openConnection>
   <ruleToSubmit>
    <event>update($a) OR insert($a)</event>
    <condition>
      FOR $a IN document( )//housestobuy/house,
      WHERE $a//cost < 1500000 AND
            contains($a//style,"victorian") AND
            contains($a//description,"furnished") AND
            $a//nr_of_bedrooms >= 4 AND
            $a//nr_of_bathrooms >= 2 AND
            $a//city="San Francisco" AND
            $a//area="Marina" AND
            empty($a//sold_to)
    </condition>
    <action>
     <soap-header>
       <uri>/NotifyHouse</uri>
       <host>housemediator.com</host>
       <soap-action>notifyHouse</soap-action>
     </soap-header>
     <SOAP-ENV:Envelope
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
                                    soap/envelope"
        SOAP-ENV:encodingStyle="http://
                 schemas.xmlsoap.org/soap/encoding">
        <SOAP-ENV:Body>
         <m:DeliverHouseNews xmlns:m="http://
                   housemediator.com/soap/methods">
          <foundthehouse>
             $a//*
          </foundthehouse>
          <server>
             www.expensivehousesincalifornia.com
          </server>
          <localHouseId>
             $a/@Id
          </localHouseId>
         </m:DeliverHouseNews>
        </SOAP-ENV:Body>
     </SOAP-ENV:Envelope>
    </action>
   </ruleToSubmit>
   <contractProposed>
     <cost>0</cost>
     <guarantee>none</guarantee>
   </contractProposed>
  </m:Subscribe>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

When a house satisfying the search is retrieved, a SOAP invocation is produced on the XML server and sent back to the Service Reseller under the format of a Rule-Generated Message.

```
POST /NotifyHouse HTTP/1.1
Host: housemediator.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "notifyHouse"

<SOAP-ENV:Envelope
 xmlns:SOAP-ENV="http://
          schemas.xmlsoap.org/soap/envelope"
```

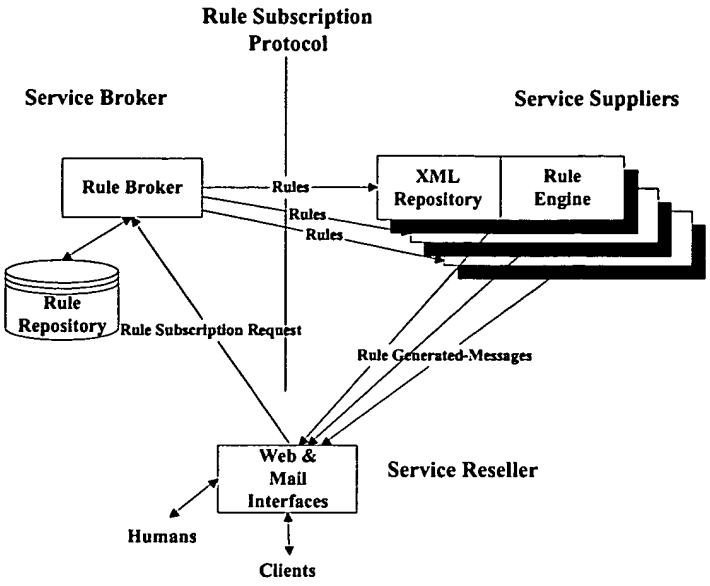


Figure 1: Reactive E-service Application Scenario

```
SOAP-ENV:encodingStyle="http://
         schemas.xmlsoap.org/soap/encoding">
<SOAP-ENV:Body>
 <m:DeliverHouseNews xmlns:m="http://
            housemediator.com/soap/methods">
  <foundthehouse>
   <house Id="1745">
     <address>
       <street>A Street</street>
       <area>Marina</area>
       <city>San Francisco</city>
       <zip>94120</zip>
     </address>
     <cost>1450000</cost>
     <squarefeet>1600</squarefeet>
     <year_construction>1925</year_construction>
     <year_refurbished>1980<year_refurbished>
     <nr_of_bedrooms>6<nr_of_bedrooms>
     <nr_of_bathrooms>2<nr_of_bathrooms>
     <nr_of_balconies>2<nr_of_balconies>
     <miscellaneous>
      <style>modern and victorian</style>
      <view>Ocean view</view>
      <description>
       Possibly furnished, wide kitchen,
       no smoking policy
      </description>
     </miscellaneous>
   </house>
  </foundthehouse>
    <server>
      www.expensivehousesincalifornia.com
    </server>
    <localHouseId>
      1745
    </localHouseId>
</m:DeliverHouseNews>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

5. ARCHITECTURAL FRAMEWORK

In this Section we introduce a proposal for the protocol used to submit rules to the repository. We are interested in presenting the features that are needed for a successful deployment of these services in the Internet context. This is the low-level protocol for the definition of e-service requests.

5.1 E-service request protocol

A minimal interface for the submission of rules is represented by the following primitives: `Connect`, `Subscribe`, `Unsubscribe`, and `Disconnect`. Our goal is to present the fundamental primitives and parameters required for a rule submission service; actual implementations will use additional support primitives and a more complex API. For readability, we represent the primitives in IDL, even if they will be implemented as SOAP invocations to the XML server. We describe separately each of the primitives.

```
ConnectionId Connect(in AuthenticatedUser user,
            in ServerProfile requestedProfile)
```

The `Connect` primitive creates a connection for the submission of rules to a remote XML repository. A connection is established, instead of using a state-less approach as in HTTP, because the submission of rules typically requires that users are authenticated and the capabilities of the server are verified. Instead of repeating these steps for every rule submission, it is preferable to do them once and separately for every set of rule submissions. Once a connection is established, all future requests originating from the same location can refer to the same connection.

The parameters of the request are an instance of *AuthenticatedUser*, which represents the user with the password or the credentials that are needed to verify his identity and

the corresponding privilege to submit rules. The second parameter is an instance of *ServerProfile*, which contains a list of features that the submitter expects to be supported by the server. For instance, the requester may present a list of rule languages and ask which are the ones accepted by the system, it may query if the server stores XML information conforming with a given DTD, or it may request if SOAP calls in the rule action should only return to the submitter; any other parameter that may be relevant in this context may be part of the server profile. If the user privileges are sufficient to open a connection with the server, the response of the call returns a valid connection identifier, combined with the answer to each of the components of the server profile.

```
SubmissionId Subscribe(in ConnectionId
                       openConnection,
                       in Rule ruleToSubmit,
                       in Contract contractProposed)
```

The `Subscribe` primitive permits the submission of a rule to a server. Its parameters are the connection which the request refers to (*openConnection*), the rule (*ruleToSubmit*), and the contract on which the service is realized (*contractProposed*). The role of the last parameter is the focus of Section 5.2. If the request is successful, it will return a valid *SubmissionId*; if the connection is not open, the rule is not understood, the contract is not accepted or the query is not satisfied, the request will not be accepted and it will return an error code, with an explanation of the motivation for the refusal in the SOAP body of the response.

```
void Unsubscribe(in ConnectionId openConnection,
                 in SubmissionId subId)
```

The `Unsubscribe` primitive is invoked when a submitted rule must be removed from a server. The request must originate from the same user who submitted the rule. The parameter *subId* is a rule identifier internal to the XML server that has been returned by the `Subscribe` request.

```
void Disconnect(in ConnectionId openConnection)
```

The `Disconnect` primitive closes the connection created by the `Connect` primitive and frees the resources required for its management on the server. A timeout mechanism can automatically invoke the primitive for connections that remain idle beyond a predefined duration.

5.2 Rule Packaging

The primitive *Subscribe* used to submit active rules to an XML Server presents a *contractProposed* parameter. This parameter specifies the contract that the rule submitter offers to the XML Server. A contract describes a set of obligations agreed by each party in the transaction. In this context, the Rule Broker will typically describe the remuneration requested to the Server for the acceptance of the rule and for its execution. Typically, a Broker will be guaranteed either a fixed rate for each installed rule, or a variable rate for each delivered message, or both. In addition, the XML Server will typically offer guarantees regarding the responsiveness of the rule, its robustness, and its availability.

Experience in the construction of B2B applications demonstrates that one of the most critical aspects for the success of a B2B initiative is the definition of adequate contracts. Even for trading commodities, where the problem of defining a market seems relatively simple, several systems failed because they were unable to represent precisely the "contract", that is, the assumption of responsibilities that a commercial transaction implies.

It is indeed quite difficult to prescribe a solution that will satisfy the requirements of every context. The main prescription that we present is the need of this component in the interface, whose exact structure depends on the characteristics of the application and requires a specific study.

6. IMPLEMENTATION

In this Section we show the guidelines that can be used for the implementation of an active rule system for XML. An interesting feature of our proposal is the relatively easy implementation and integration with existing Web solutions, based on the reuse of several current Web standards and of their robust implementations.

The run-time behavior of a single rule execution is modeled by a process that binds the three parts of the rule, interacting with the subsystems responsible for each of them: the DOM Event Model, the XQuery engine, and the generator of SOAP calls.

The DOM Event Model permits the definition of arbitrary event listeners, which consist of generic procedures, written in the language implementing the interface (e.g., Java), that need to present the implementation of a set of predefined methods. The event listener can be dynamically associated with the document nodes, for a certain event, and its methods are invoked when an event of the specified kind on the specified node is generated. As we discuss below, the Event Model of the DOM generates processes for rule execution according to two different alternatives. In either case, the rule process first extracts from the event the information that may be required to complete the XQuery query that represents the condition. Then, the condition is executed by a XQuery processor, which may return an empty result or an XML fragment. When the result is empty, the condition is not satisfied, and the execution of the rule process terminates. If the query returns an XML fragment, then the rule action is executed, by extracting from the query result the information necessary for the construction of the SOAP call; then, the call is submitted to the server appearing in one of the action elements.

The rule engine is the software component at the core of an active rule system. In database trigger systems, its responsibilities include the determination of the triggered rules, the selection of the order of rule execution among rules triggered at the same time, and in general the coordination among separate rules [5]. This role is greatly simplified by the assumptions on rules that we made in this paper.

We have seen the detection of events and the triggering of rules is directly managed by the Event Model of the DOM. We identify two main alternatives for the realization of this task.

- The *centralized* solution associates a single event listener with the root of the document. This alternative requires the creation of a complex event listener, which can be considered as the main component of the rule engine. The event listener classifies each event and detects if the occurred event triggers one of the ac-

tive rules. This task requires an efficient description of the rule repository, with a structure similar to that required for the implementation of trigger engines. Once a rule is triggered, the process managing it is activated.

- The *fragmented* solution creates a set of event listeners, associated with every node instance on which events have to be monitored. This solution introduces an event listener for every rule, and associates the event listener with every node instance on which the triggering events can occur.

In the database field, previous research has analyzed the impact that various design parameters could have on system performance [4]. Based on those results and on the characteristics of this context, it is possible to estimate that the main criteria to use in the choice between the centralized and the fragmented solution are the cost of the association between event listeners and nodes, the number of monitored nodes compared with the number of document nodes, and the frequency of irrelevant events. The fragmented solution is preferable when the association between an event listener and a node requires limited resources, when the monitored nodes are a small fraction of the document nodes, and when the events that trigger rules are very few compared with the produced events.

A particularly sophisticated implementation of the fragmented solution may be optimized to restrict the creation of event listeners only to the nodes that have the potential to satisfy the rule condition. This strategy has a great potential and is the one that can offer the best performance in many applications.

7. CONCLUSIONS

In this paper, we propose the use of active rules for pushing reactive services; we have shown that such services satisfy the needs of many important applications. Below, we list some of the possible obstacles that could limit the applicability of our solution, and explain why each of them is not critical.

The first observation is that in order to write an efficient rule it may be necessary to know the schema of XML resources; this is a serious obstacle for a wide-scale rule deployment, where the Rule Broker submits the rule to many different and autonomous sites. Indeed, this problem is faced by most of the wide-scale, inter-business applications based on XML. Fortunately, many ongoing initiatives (RosettaNet [40], eCo Framework [26], OMG CORBA [18], ebXML [25]) are dedicated to the definition of DTDs and schemas for specific industrial sectors, that should permit interoperability among systems. These efforts will offer considerable benefits to our application as well.

Another observation is that a rule is an external program coming from an external system, whose execution may present unacceptable risks (such as: access to protected resources, malicious code, etc.). This is an intrinsic characteristic of any agent-based mechanism. In our context, however, rules are installed only by trusted sites (the Rule Brokers) and they are (on purpose) severely restricted in the scope of their actions. They cannot modify the state of XML resources, and SOAP calls can be addressed only to given, certified methods. Thus, we believe that the application scenario considered in this paper is sufficiently protected and secure.

A final observation is that most relational trigger systems and applications do not exhibit high scalability: when the number of triggers becomes large, applications often become inefficient and unmanageable. However, from the description of the implementation given in Section 6, it is evident that the rule system hereby proposed is much simpler than a generic trigger engine. In particular, interactions among rules are excluded, thereby eliminating one of the main causes of inefficiency and mismanagement. In addition, knowing in advance the structure of rules gives great potential for optimization strategies. In any case, the transition from generic, trigger-based implementations of services (enabling their rapid prototyping) to efficient embedded solutions has already occurred in many active rule applications [9], and could take place in our context as well.

In summary, services based on reactive push technology have great potential applicability; their flexibility and ease of implementation can make such services one of the key ingredients of future Web-based infrastructures.

8. REFERENCES

[1] S. Abiteboul, C. Cluet, L. Mignet, B. Amann, T. Milo, and A. Eyal. "Active Views for Electronic Commerce". In *25th Very Large Data Bases Conference Proceedings*, pages 138–149, September 1999.

[2] M. Altinel and M. Franklin. "Efficient Filtering of XML Documents for Selective Dissemination of Information". In *26th Very Large Data Bases Conference Proceedings*, pages 53–64, September 2000.

[3] "Amazon Web Site", 2001. http://www.amazon.com.

[4] E. Baralis and A. Bianco. "Performance Evaluation of Rule Semantics in Active Databases". In *13th ICDE Conference Proceedings*, pages 365–374, April 1997.

[5] E. Baralis, S. Ceri, and S. Paraboschi. "Compile-Time and Runtime Analysis of Active Behaviors". *IEEE Transactions on Knowledge and Data Engineering*, 10(3):353–370, 1998.

[6] A. Bonifati, S. Ceri, and S. Paraboschi. "Active Rules for XML: A New Paradigm for E-Services". In *1st Workshop on E-Services (co-held with the 26th Very Large Data Bases Conference) Proceedings*, September 2000.

[7] T. Bray, J. Paoli, and C. M. Sperberg-McQueen (Eds.). "Extensible Markup Language (XML) 1.0-2nd Edition", Oct. 2000. http://www.w3.org/TR/2000/REC-xml-20001006.

[8] "Broadvision Web Site", 2001. http://www.broadvision.com.

[9] S. Ceri, R. J. Cochrane, and J. Widom. "Practical Applications of Triggers and Constraints: Success Stories and Lingering Issues". In *26th Very Large Data Bases Conference Proceedings*, pages 254–262, September 2000.

[10] S. Ceri, S. Comai, E. Damiani, P. Fraternali, S. Paraboschi, and L. Tanca. "XML-GL: a Graphical Language for Querying and Restructuring WWW Data". In *WWW8 Conference Proceedings*, May 1999.

[11] S. Ceri and G. Pelagatti. *"Distributed Databases: Principles and Systems"*. McGraw Hill, 1984.

[12] S. Ceri and J. Widom. "Production Rules in Parallel and Distributed Database Environments". In *18th Very Large Data Bases Conference Proceedings*, August 1992.

[13] S. Chakravarthy, R. Le, and R. Dasari. "ECA Rule Processing in Distributed and Heterogeneous Environments". In *International Symposium on Distributed Objects and Applications*, 1999.

[14] D. Chamberlin, J. Robie, and D. Florescu. "Quilt: An XML Query Language for Heterogeneous Data Sources". In *Webdb Conference Proceedings*, Dallas, TX, May 2000.

[15] D. Chamberlin et al.(Eds.). "XQuery: A Query Language for XML. W3C Working Draft, 15 February 2001.". http://www.w3.org/TR/2001/WD-xquery-20010215/.

[16] J. Chen, D. DeWitt, F. Tian, and Y. Wang. "NiagaraCQ: A Scalable Continuous Query System for Internet Databases". In *ACM Sigmod Conference Proceedings*, May 2000.

[17] J. Clark (Eds.). "XML Transformations (XSLT) Version 1.0", Nov. 1999. http://www.w3.org/TR/xslt.

[18] "Corba Web Site", 2001. http://www.omg.org.

[19] "Crossgain Web Site", 2001. http://www.crossgain.com.

[20] G. Cugola, E. D. Nitto, and A. Fuggetta. "Exploiting an Event-Based Infrastructure to Develop Complex Distributed Systems". In *20th International Conference on Software Engineering Proceedings*, April 1998.

[21] A. Deutsch, M. F. Fernandez, D. Florescu, A. Y. Levy, and D. Suciu. "A Query Language for XML". In *WWW8 Conference Proceedings*, Toronto, Canada, May 1999. http://www8.org/fullpaper.html.

[22] "Document Object Model (DOM) Level 2 Core Specification Version 1.0 W3C Proposed Recommendation", September 2000. http://www.w3.org/TR/2000/PR-DOM-Level-2-Core-20000927/.

[23] "Dynamo Web Site", 2001. http://www.dynamo-ny.com/flash_index.html.

[24] "E-Speak Web Site", 2001. http://www.e-speak.hp.com/map.shtm.

[25] "ebXML Web Site", 2001. http://www.ebXML.org.

[26] "The eCo Framework", 2001. http://www.commerce.net.

[27] P. Fankhauser et al.(Eds). "The XML Query Algebra", Dec. 2000. http://www.w3.org/TR/query-algebra/.

[28] S. Gatziu, A. Koschel, G. von Bultzingsloewen, and H. Fritschi. "Unbundling Active Functionality". *ACM Sigmod Record*, 27(1):35–40, 1998.

[29] R. Gruber, B. Krishnamurthy, and E. Panagos. "The Architecture of the READY Event Notification Service". In *Workshop on Middleware (co-held with the International Conference on Distributed Computing Systems) Proceedings*, May 1999.

[30] E. N. Hanson, C. Carnes, L. Huang, M. Konyola, L. Norohna, S. Parthasarathy, and J. B. Park. "Scalable Trigger Processing". In *15th ICDE Conference Proceedings*, pages 266–275, 1999.

[31] "ICE Protocol Specification (CGA)", 1999. http://www.idealliance.org/ice/ice_note-ice-19990519.asp.

[32] "Internet Object Transfer Protocol Specification (Commerce One)", 2001. http://search.ietf.org/internet-drafts/draft-ietf-trade-iotp-v1.0-protocol-07.txt.

[33] "J. McHugh and S. Abiteboul and R. Goldman and D. Quass and J. Widom". Lore: A database management system for semistructured data. *ACM Sigmod Record*, 26(3):54–66, September 1997.

[34] A. Koschel, S. Gatziu, H. Fritschi, and G. von Bultzingsloewen. "Applying the Unbundling Process To Active Database Systems". In *International Workshop on Issues and Applications of Database Technology Proceedings*, Toronto, Canada, July 1998.

[35] B. Krishnamurthy and D. Rosemblum. "Yeast: A General Purpose Event-Action System". *IEEE Transactions on Software Engineering*, 21(10), 1995.

[36] Object Management Group 2000. "Event Service Specification", 2000. http://www.omg.org/technology/documents/formal/event_service.htm.

[37] Object Management Group 2000. "Notification Service Specification", 2000. http://www.omg.org/technology/documents/formal/notification_service.htm.

[38] "Propel Web Site", 2001. http://www.propel.com.

[39] J. Robie (Ed.). "XQL (XML Query Language)", Aug. 1999. http://www.ibiblio.org/xql/xql-proposal.html.

[40] "RosettaNet Framework", 2001. http://www.rosettanet.org.

[41] "Simple Object Access Protocol (SOAP) 1.1 (W3C Note)", 2000. http://www.w3.org/TR/2000/NOTE-SOAP-20000508/.

[42] "System Management Arts. Root Cause Analysis and its Role in Event Correlation (White Paper)", 2001. http://www.smarts.com.

[43] "Tamino Product Web Site", 2000. http://www.softwareag.com/tamino/.

[44] G. von Bultzingsloewen, A. Koschel, P. Lockemann, and H. Walter. "ECA Functionality in a Distributed Environment". *In Norman W. Paton (ed),* Active Rules in Database Systems, pages 147–175, 1999.

[45] J. Widom. "Research Problems in Data Warehousing". In *Information and Knowledge Management Conference Proceedings*, pages 583–595, August 1995.

[46] "XML Metadata Interchange (Unysis et al.)". Updated 2 October 1999. http://www.omg.org/cgi-bin/doc?ad/99-10-02.

[47] "XML Protocol Requirements. W3C Working Draft 19 December 2000 ". http://www.w3.org/TR/2000/WD-xp-reqs-20001219/.

[48] "XML-RPC Specification (Userland)". Updated 16 October 1999. http://www.xmlrpc.com/spec.

[49] "Yahoo Web Site", 2000. http://www.yahoo.com.

[50] G. Zhou, R. Hull, R. King, and J. Franchitti. "Supporting Data Integration and Warehousing Using H20". *IEEE Data Engineering Bulletin*, 18(2):29–40, June 1995.

ACM PORTAL THE ACM DIGITAL LIBRARY

> home > about > feedback > login

US Patent & Trademark Office

Try the *new* Portal design

Give us your opinion after using it.

Citation

International World Wide Web Conference >archive

Proceedings of the tenth international conference on World Wide Web >toc

2001 , Hong Kong, Hong Kong

Pushing reactive services to XML repositories using active rules

Authors

Angela Bonifati

Stefano Ceri

Stefano Paraboschi

Sponsors

SIGWEB : ACM Special Interest Group on Hypertext, Hypermedia, and Web

SIGLINK : Hypertext, Hypermedia, and Web

IW3C2 : International World Wide Web Conference Committee

Publisher

ACM Press New York, NY, USA

Pages: 633 - 641 Series-Proceeding-Article

Year of Publication: 2001

ISBN:1-58113-348-0

doi> http://doi.acm.org/10.1145/371920.372168 (Use this link to Bookmark this page)

> full text > references > citings > index terms > peer to peer

> Discuss > Similar > Review this Article Save to Binder

> BibTex Format

↑ **FULL TEXT:** Access Rules

pdf 204 KB

↑ **REFERENCES**

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

1 S. Abiteboul, C. Cluet, L. Mignet, B. Amann, T. Milo, and A. Eyal. "Active Views for Electronic Commerce". In 25th Very Large Data Bases Conference Proceedings, pages 138-149, September

1999.

2 M. Altinel and M. Franklin. "Efficient Filtering of XML Documents for Selective Dissemination of Information". In 26th Very Large Data Bases Conference Proceedings, pages 53-64, September 2000.

3 "Amazon Web Site", 2001. http://www.amazon.com.

4 E. Baralis and A. Bianco. "Performance Evaluation of Rule Semantics in Active Databases". In 13th ICDE Conference Proceedings, pages 365-374, April 1997.

5 E. Baralis, S. Ceri, and S. Paraboschi. "Compile-Time and Runtime Analysis of Active Behaviors". IEEE Transactions on Knowledge and Data Engineering, 10(3):353-370, 1998.

6 A. Bonifati, S. Ceri, and S. Paraboschi. "Active Rules for XML: A New Paradigm for E-Services". In 1st Workshop on E-Services (co-held with the 26th Very Large Data Bases Conference) Proceedings, September 2000.

7 T. Bray, J. Paoli, and C. M. Sperberg-McQueen (Eds.). "Extensible Markup Language (XML) 1.0-2nd Edition", Oct. 2000. http://www.w3.org/TR/2000/REC-xml-20001006.

8 "Broadvision Web Site", 2001. http://www.broadvision.com.

9 S.Ceri, R. J. Cochrane, and J. Widom. "Practical Applications of Triggers and Constraints: Success Stories and Lingering Issues". In 26th Very Large Data Bases Conference Proceedings, pages 254-262, September 2000.

10 Stefano Ceri , Sara Comai , Ernesto Damiani , Piero Fraternali , Stefano Paraboschi , Letizia Tanca, XML-GL: a graphical language for querying and restructuring XML documents, Proceeding of the eighth international conference on World Wide Web, p.1171-1187, May 1999, Toronto, Canada

11 Stefano Ceri , Giuseppe Pelagatti, Distributed databases principles and systems, McGraw-Hill, Inc., New York, NY, 1984

12 S. Ceri and J. Widom. "Production Rules in Parallel and Distributed Database Environments". In 18th Very Large Data Bases Conference Proceedings, August 1992.

13 S. Chakravarthy, R. Le, and R. Dasari. "ECA Rule Processing in Distributed and Heterogeneous Environments". In International Symposium on Distributed Objects and Applications, 1999.

14 D. Chamberlin, J. Robie, and D. Florescu. "Quilt: An XML Query Language for Heterogeneous Data Sources". In Webdb Conference Proceedings, Dallas, TX, May 2000.

15 D. Chamberlin et al.(Eds.). "XQuery: A Query Language for XML. W3C Working Draft, 15 February 2001.". http://www.w3.org/TR/2001/WD-xquery-20010215/.

16 Jianjun Chen , David J. DeWitt , Feng Tian , Yuan Wang, NiagaraCQ: a scalable continuous query system for Internet databases, Proceedings of the 2000 ACM SIGMOD international conference on Management of data, p.379-390, May 15-18, 2000, Dallas, Texas, United States

17 J. Clark (Eds.). "XML Transformations (XSLT) Version 1.0", Nov. 1999. http://www.w3.org/TR/xslt.

18 "Corba Web Site", 2001. http://www.omg.org.

19 "Crossgain Web Site", 2001. http://www.crossgain.com.

20 G. Cugola , E. Di Nitto , A. Fuggetta, Exploiting an event-based infrastructure to develop complex distributed systems, Proceedings of the 20th international conference on Software engineering, p.261-270, April 19-25, 1998, Kyoto, Japan

21 Alin Deutsch , Mary Fernandez , Daniela Florescu , Alon Levy , Dan Suciu, A query language for XML, Proceeding of the eighth international conference on World Wide Web, p.1155-1169, May 1999, Toronto, Canada

22 "Document Object Model (DOM) Level 2 Core Specification Version 1.0 W3C Proposed Recommendation", September 2000. http://www.w3.org/TR/2000/PR-DOM-Level-2-Core-20000927/.

23 "Dynamo Web Site", 2001. http://www.dynamo-ny.com/ ash index.html.

24 "E-Speak Web Site", 2001. http://www.e-speak.hp.com/map.shtm.

25 "ebXML Web Site", 2001. http://www.ebXML.org.

26 "The eCo Framework", 2001. http://www.commerce.net.

27 P. Fankhauser et al.(Eds). "The XML Query Algebra", Dec. 2000. http://www.w3.org/TR/query-algebra/.

28 Stella Gatziu , Arne Koschel , Günter von Bültzingsloewen , Hans Fritschi, Unbundling active functionality, ACM SIGMOD Record, v.27 n.1, p.35-40, March 1998

29 R. Gruber, B. Krishnamurthy, andE.Panagos. "The Architecture of the READY Event Notification Service". In Workshop on Middleware (co-held with the International Conference on Distributed Computing Systems) Proceedings, May 1999.

30 E. N. Hanson, C. Carnes, L. Huang, M. Konyola, L. Norohna, S. Parthasarathy, and J. B. Park. "Scalable Trigger Processing". In 15th ICDE Conference Proceedings, pages 266-275, 1999.

31 "ICE Protocol Specification (CGA)", 1999. http://www.idealliance.org/ice/ice note-ice-19990519.asp.

32 "Internet Object Transfer Protocol Specification (Commerce One)", 2001. http://search.ietf.org/internet-drafts/draft-ietf-tradeiotp-v1.0-protocol-07.txt.

33 Jason McHugh , Serge Abiteboul , Roy Goldman , Dallas Quass , Jennifer Widom, Lore: a database management system for semistructured data, ACM SIGMOD Record, v.26 n.3, p.54-66, Sept. 1997

34 A. Koschel, S. Gatziu, H. Fritschi, and G. von Bultzingsloewen. "Applying the Unbundling Process To Active Database Systems". In International Workshop on Issues and Applications of Database Technology Proceedings, Toronto, Canada, July 1998.

35 B. Krishnamurthy and D. Rosemblum. "Yeast: A General Purpose Event-Action System". IEEE Transactions on Software Engineering, 21(10), 1995.

36 Object Management Group 2000. "Event Service Specification", 2000. http://www.omg.org/technology/documents/formal/ event service.htm.

37 Object Management Group 2000. "Notification Service Specification", 2000. http://www.omg.org/technology/documents/formal/ notification service.htm.

38 "Propel Web Site", 2001. http://www.propel.com.

39 J. Robie (Ed.). "XQL (XML Query Language)", Aug. 1999. http://www.ibiblio.org/xql/xql-proposal.html.

40 "RosettaNet Framework", 2001. http://www.rosettanet.org.

41 "Simple Object Access Protocol (SOAP) 1.1 (W3C Note)", 2000. http://www.w3.org/TR/2000/NOTE- SOAP-20000508/.

42 "System Management Arts. Root Cause Analysis and its Role in Event Correlation (White Paper)", 2001. http://www.smarts.com.

43 "Tamino Product Web Site", 2000. http://www.softwareag.com/tamino/.

44 G. von Bultzingsloewen, A. Koschel, P. Lockemann, and H. Walter. "ECA Functionality in a Distributed Environment". In Norman W. Paton (ed), Active Rules in Database Systems, pages 147-175, 1999.

45 Jennifer Widom, Research problems in data warehousing, Proceedings of the fourth international conference on Information and knowledge management, p.25-30, November 29-December 02, 1995, Baltimore, Maryland, United States

46 "XML Metadata Interchange (Unysis et al.)". Updated 2 October 1999. http://www.omg.org/cgi-bin/doc?ad/99-10-02.

47 "XML Protocol Requirements. W3C Working Draft 19 December 2000 ". http://www.w3.org/TR/2000/WD-xp-reqs-20001219/.

48 "XML-RPC Specification (Userland)". Updated 16 October 1999. http://www.xmlrpc.com/spec.

49 "Yahoo Web Site", 2000. http://www.yahoo.com.

50 G. Zhou, R. Hull, R. King, and J. Franchitti. "Supporting Data Integration and Warehousing Using H20". IEEE Data Engineering Bulletin, 18(2):29-40, June 1995.

↑ CITINGS *2*

James Bailey , Alexandra Poulovassilis , Peter T. Wood, An event-condition-action language for XML, Proceedings of the eleventh international conference on World Wide Web, May 07-11, 2002, Honolulu, Hawaii, USA

Juan M. Ale , Mauricio Minuto Espil, Active rules and active databases: concepts and applications, Effective databases for text & document management, Idea Group Publishing, Hershey, PA, 2003

↑ INDEX TERMS

Primary Classification:
H. Information Systems

Additional Classification:

H. Information Systems

↳ **H.3** INFORMATION STORAGE AND RETRIEVAL

↳ **H.3.5** On-line Information Services

↳ **Subjects:** Web-based services

↳ **H.5** INFORMATION INTERFACES AND PRESENTATION (I.7)

↳ **H.5.3** Group and Organization Interfaces

↳ **Nouns:** WEB

I. Computing Methodologies

↳ **I.7** DOCUMENT AND TEXT PROCESSING

General Terms:
Languages, Management, Performance, Theory

Keywords:
SOAP, XML, active rules, document management, push technology, query languages for XML

↑ **Peer to Peer - Readers of this Article have also read:**

- M⁴: a metamodel for data preprocessing
  Proceedings of the 4th ACM international workshop on Data warehousing and OLAP
  Anca Vaduva , Jörg-Uwe Kietz , Regina Zücker
- Data structures for quadtree approximation and compression
  Communications of the ACM 28, 9
  Hanan Samet
- A hierarchical single-key-lock access control using the Chinese remainder theorem
  Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing
  Kim S. Lee , Huizhu Lu , D. D. Fisher
- 3D representations for software visualization
  Proceedings of the 2003 ACM symposium on Software visualization
  Andrian Marcus , Louis Feng , Jonathan I. Maletic
- Probabilistic surfaces: point based primitives to show surface uncertainty
  Proceedings of the conference on Visualization '02
  Gevorg Grigoryan , Penny Rheingans

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

A Predicate Matching Algorithm for Database Rule Systems*

Eric N Hanson[1,2] *Moez Chaabouni*[2]
Chang-Ho Kim[2] *Yu-Wang Wang*[2]

[1] USAF Wright R & D Center
WRDC/TXI
Dayton, OH 45433

[2] Wright State University
Dept of Computer Science
Dayton, OH 45435

Abstract

Forward-chaining rule systems must test each newly asserted fact against a collection of predicates to find those rules that match the fact Expert system rule engines use a simple combination of hashing and sequential search for this matching We introduce an algorithm for finding the matching predicates that is more efficient than the standard algorithm when the number of predicates is large We focus on equality and inequality predicates on totally ordered domains This algorithm is well-suited for database rule systems, where predicate-testing speed is critical A key component of the algorithm is the *interval binary search tree* (IBS-tree) The IBS-tree is designed to allow efficient retrieval of all intervals (e g range predicates) that overlap a point, while allowing dynamic insertion and deletion of intervals The algorithm could also be used to improve the performance of forward-chaining inference engines for large expert systems applications

1 Introduction

Efficient testing of rule predicates is critical for good performance of forward-chaining rule systems Extensive research has been done on processing rule conditions efficiently, including development of the Rete algorithm [For82], a modified version of Rete called TREAT [Mir87], and extensions to the Rete algorithm to exploit parallelism [KS89] In this paper, we investigate an important part of the rule condition testing problem testing a collection of predicates to see which of the predicates match a single fact In database terminology, which we will use hereafter, this is the problem of testing a tuple to see which of a collection of single-relation selection conditions match the tuple

This paper does not address the issue of how join predicates will be processed Efficient ways to determine which single-relation selection predicates match every new and modified tuple are important because this match must be done regardless of how the join clauses of rule conditions are tested

The predicate testing problem in a database rule system is defined as follows We are given a database containing a set of n relations, R_1 R_n, and m production rules (triggers), r_1 r_m Rules are of the form

if *condition*
then *action*

A rule condition can be an expression containing a conjunction of selection conditions and joins (projection is not allowed in rule conditions) Considering only the selection conditions of the rules, there is a collection of k single-relation predicates, P_i, $1 \leq i \leq k$ Each predicate restricts one or more attributes of a tuple t from a relation R_j where $1 \leq j \leq n$ We assume that any predicate containing a disjunction is broken up into two or more predicates that do not have disjunction, and these predicates are treated separately The general form of a predicate for purposes of this discussion is a conjunction of the following form

$$P_i \equiv (\text{the tuple } t \text{ is in relation } R_j) \wedge C_1 \wedge C_2 \wedge \quad C_q$$

where each C_j, $1 \leq j \leq q$, is one of the following

$C_j \equiv const_1\ \rho_1\ t$ attribute $\rho_2\ const_2$
$C_j \equiv t$ attribute $= const_1$
$C_j \equiv$ function(t attribute)

In addition, $const_1 \leq const_2$, both $const_1$ and $const_2$ are drawn from the domain of legal values for t attribute, and ρ_1 and ρ_2 are one of $\{<, \leq\}$ Equality predicates are a special case of interval predicates, but since they are so common, they are listed separately Open intervals are specified by setting $const_1$ or $const_2$ to $-\infty$ or $+\infty$, respectively For predicate clauses of the form "function(t attribute)," nothing is assumed about the function except that it returns true or false

Some example predicates will be shown below, based on this relation schema

EMP(name, age, salary, dept)

Here are some examples of predicates on tuples of the relation EMP

EMP salary < 20000 and EMP age > 50
20000 ≤ EMP salary ≤ 30000
EMP job = "Salesperson"
IsOdd(EMP age) and EMP dept ="Shoe"

In the last predicate above, IsOdd is a function that returns true if its argument is an odd number, and false otherwise

Given the collection of predicates described above, and a tuple t, the predicate testing problem is to determine exactly those P_i's that match t One approach to testing predicates is to use a predicate index Many approaches to the predicate indexing problem have been developed In Section 2 we discuss these methods in order of increasing complexity Section 3 covers pragmatic considerations for predicate indexing regarding characteristics of data in relational database applications, and the expected characteristics of rules Section 4 introduces an alternate predicate indexing algorithm called the *interval binary search tree* (IBS-tree) which we argue is

*This work was supported by the Air Force Office of Scientific Research under grant number AFOSR-89-0286

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery To copy otherwise, or to republish, requires a fee and/or specific permission
© 1990 ACM 089791 365 5/90/0005/0271 $1 50

readily implementable, and should perform well in realistic database rule processing applications Section 5 presents an analysis that shows the performance characteristics of the method described in Section 4 Finally, Section 6 summarizes and presents conclusions

2 Review of Predicate Indexing Methods

Predicate indexing methods range from simple sequential testing to use of complex geometric data structures Below we list some alternatives that have been proposed for predicate indexing in a DBMS, in order of increasing complexity Parallel predicate indexing methods are not considered since our primary focus is a fast uniprocessor implementation For each method, we discuss what is done when a tuple is modified or inserted into the DBMS

2.1 Sequential Search

In this method, the system traverses a list of predicates sequentially, testing each against the tuple This has low overhead and works well for small numbers of predicates, but clearly performs badly when the number of predicates is large

2.2 Hash on Relation Name Plus Sequential Search

In this method, the system maintains one list of predicates for each relation, and for each tuple modified, hashes on relation name to locate the predicate list for the tuple The predicates on the list are then tested against the tuple sequentially This is essentially the algorithm used in many main-memory-based production rule systems including some implementations of OPS5 [For81, Mir87] The algorithm performs well when the average number of predicates per relation is small, and the predicates are distributed evenly over the relations

2.3 Physical Locking

This method, discussed in [SSH86, SHP88], involves treating a predicate clause like a query, and running the standard query optimizer to produce an access plan for the query [S*79] If the resulting access plan requires an index scan, then special persistent markers (locks) are placed on all tuples read during the scan, and all index intervals inspected during the scan If the resulting access plan is a sequential search, then "lock escalation" is performed, and a relation-level lock is placed on the relation being scanned When a tuple is modified or inserted, the system collects locks that conflict with the update (i e all relation level locks, any locks that conflict with any indexes that were updated, and any other locks previously on the tuple) For each of the locks collected, the system tests the tuple against the predicate associated with the lock

This algorithm has the advantage that no main-memory is needed to hold a predicate index, so theoretically, a very large number of rules can be accommodated In addition, the algorithm makes use of the standard indexes and query processor to index predicates However, there are disadvantages to the approach In particular, when there are no indexes, or a large number of predicate clauses lie on attributes which do not have an index, most predicates will have a relation-level lock This degenerate case requires sequentially testing a new or modified tuple against all the predicates for a particular relation, resulting in bad worst-case performance when the number of predicates is large Also, the set of predicates must be stored in main memory to avoid costly disk I/O to test a tuple against a predicate when a lock for that predicate is found This negates some of the memory-saving advantages of the algorithm In addition, the need to set locks on index intervals and on tuples complicates the implementation of storage structures

2.4 Multi-dimensional indexing

This technique stores a collection of predicates in a multi-dimensional structure designed for indexing region data Applicable indexes include the R-tree [Gut84] and R+-tree [SSH86] Predicates are treated as regions in a k-dimensional space (where k is the number of attributes in the relation on which the predicates are defined), and inserted into a k-dimensional index Each new or modified tuple is used as a key to search the index to find all predicates that "overlap" the tuple This technique works well when most predicates are small closed regions in the space defined by the schema of the relation from which tuples are drawn However, real relational database applications often involve relations with anywhere from one to over 100 attributes, with a large fraction of relations having from 5 to 25 attributes Typical predicates on these relations (e g single-relation selection conditions in WHERE clauses of queries) normally refer to only one or two attributes, and rarely more than five [Col89] Collections of low dimension predicates like these are not small closed regions Rather, they are "slices" through space that overlap extensively Spatial data structures, particularly R-trees and R+-trees, index regions like these poorly, giving slow search performance

3 Practical Considerations for Predicate Indexing in a DBMS

Numerous database rule systems have been proposed recently, including Ariel [Han89], RPL [DE88], the POSTGRES rules system [SHP88], HiPAC [DBB*88], and DIPS [SLR89] We envision that applications built using systems like these will be primarily data management applications, enhanced with rules to provide improved data integrity, monitoring capability, and some features similar to those found in expert systems

Database rule system applications will have to handle large volumes of data (perhaps millions of records) However, we expect that the number of rules in the majority of database rule system applications will be small enough that the set of rules and data structures for rule condition testing will be small enough to fit in main memory We believe that this assumption is reasonable because rules are a form of intentional data (schema) as opposed to extensional data (contents) Moreover, the largest expert system applications built to date have on the order of 10,000 rules [BO89], which is few enough that data structures associated with the rules will fit in a few megabytes of main memory More typical rule-based system applications have on the order of 50 to 1000 rules

It is possible to concoct hypothetical applications where a tremendous number of rules are used, more than can fit in a

main-memory data structure Normally, rules in such applications have a very regular structure This regular structure can be exploited to redesign the application so that only a few rules are used in conjunction with a much larger data table The rules then use pattern matching to extract data from the table For example, consider an application for stock reordering in a grocery store The store might have 50,000 items for sale, with a relation ITEMS containing one tuple for each item One way to implement the application would be to have one rule for each item to test whether the stock of the item is below a re-order threshold An alternative way to implement the application would be to add a field to the ITEMS table containing the re-order threshold, and a single rule which compares the current stock level to the re-order stock level This second implementation is clearly preferable

It is standard practice in programming expert systems to put as much of the knowledge as possible into "facts" (e g frames or tuples) and as little as possible into rules This is done because knowledge structures are more regular and easier to understand than rules This practice will be even more important in database rule system applications, where most of the "knowledge" should be stored in the database, with minimal use of rules

The above discussion is a partial justification for building a carefully tuned main-memory predicate index to test selection predicates of rules We discuss such a predicate index in the next section

4 A High-Performance Predicate Indexing Method

In this section we introduce a predicate indexing method tailored to the problem of testing rule selection conditions in a database rule system The task the algorithm must perform is, given a set of single-relation selection predicates as described earlier, be able to return a list of all the predicates that match a tuple t from a relation R We want the algorithm to have the following properties

1 the ability to support general selection predicates composed of a conjunction of clauses on one or more attributes of a relation,

2 fast predicate matching performance,

3 the ability to rapidly insert and delete predicates online

In the algorithm we propose, the system builds an index which has at the top level a hash table, using relation names as keys, similar to high-performance implementations of production systems mentioned previously Each entry in the table contains a pointer to a second-level index for each relation This index maintains a list of non-indexable predicates In addition, the second-level index contains a set of one-dimensional indexes, one for each attribute of the relation for which one or more indexable predicate clauses have been defined This one-dimensional index is a balanced IBS-tree which allows efficient searching to determine which interval and equality predicates match a value For predicates that are a conjunction of selection clauses, if there is an indexable clause, the most selective one is placed in the IBS-tree (selectivity estimates are obtained from the query optimizer) A diagram for this indexing scheme is shown in Figure 1 In addition to this index, there is a main-memory table called PREDICATES that holds the predicates When a partial match between a tuple t and a predicate P is found, P is retrieved from PREDICATES and tested against t to see if there is a complete match Below, we focus in more detail on the IBS-tree

4.1 Dynamic Indexing of Intervals

The IBS-tree was motivated by the need to efficiently find all points, intervals, and open-ended intervals that match a particular query value in a dynamic environment where predicates can be added and deleted on-line Data structures for indexing intervals in a static environment where all intervals are known in advance include *segment trees* and *interval trees* [Sam88, Sam90] In the database rule system environment, segment trees and interval trees are not adequate because they do not allow dynamic insertion and deletion of predicates

A data structure that can index intervals dynamically is the *priority search tree* [McC85] An advantage of the priority search tree over the IBS-tree is the priority search tree requires only $O(N)$ space to index N intervals, while as we shall see in Section 4, the IBS-tree requires $O(N \log N)$ space in the worst case (but $O(N)$ in the best case) However, the priority search tree appears more complex to implement than the IBS-tree Moreover, the IBS-tree can directly accommodate multiple intervals with the same lower bound, which the interval tree cannot do To handle intervals with the same lower bound, priority search trees must use a special transformation from pairs with non-unique lower bounds to pairs with unique lower bounds This transformation is not trivial, and it must be created for each different data type to be indexed In contrast, IBS-trees work without modification on any totally ordered domain for which the comparison operators $\{<, =, >\}$ are defined — no additional code is needed

One-dimensional R-trees can also index intervals dynamically [Gut84] However, due to their generality, and the indexing heuristics required, R-trees are challenging to implement R-trees also require only $O(N)$ space Their performance should be good for intervals with low overlap, but when there is heavy overlap, search performance can worsen significantly Also, R-trees cannot accommodate open intervals

Below, we show how a binary search tree can be augmented to index intervals, resulting in the IBS-tree Then, we discuss extensions that allow the tree to remain dynamically balanced

4.2 Interval Binary Search Trees

In this section we introduce a method for augmenting a binary search tree with additional information to make it possible to rapidly find all intervals that overlap a point The IBS-tree can accommodate points and open intervals as well as closed intervals Nodes in the tree have the following form

Value	a data value representing the end point of an interval or the constant in an equality predicate
>	a set of interval identifiers
=	a set of interval identifiers
<	a set of interval identifiers
left	subtree holding all nodes with values less then Value
right	subtree holding all nodes with values greater than Value

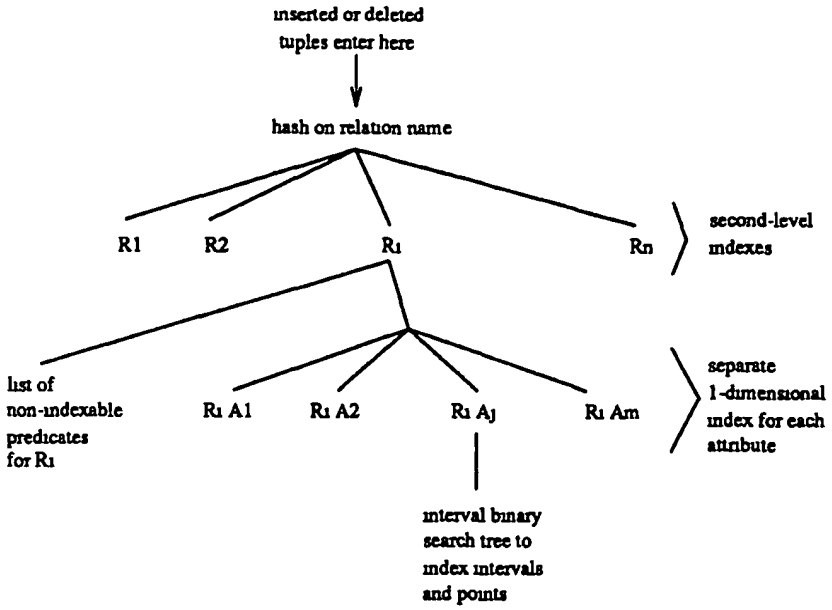


Figure 1 High-level diagram of predicate indexing scheme

The Value, left, and right entries of these nodes mean the same as they do in a standard binary search tree The "=" slot of each node contains a set of interval identifiers If an interval identifier I is contained in the = slot of a node with value V, that implies that I overlaps V The "<" slot also contains a set of interval identifiers If an interval identifier I appears in the < slot of a node with value V, that implies that any value X *that would be inserted into the left subtree of* V lies within interval I The meaning of the ">" slot is symmetric to that of the < slot

Nodes of an IBS-tree are represented graphicly using four boxes organized in the shape of an upside-down "T" The upper box contains the value for the node The lower three boxes contain the <, = and > sets, ordered from left to right An example set of intervals and the IBS-tree for those intervals is shown in Figure 2

Let P denote an interval predicate, and let PC_l and PC_r be the left and right boundaries of P, respectively Let $P\rho_l$ and $P\rho_r$ denote the comparison relations (one of $<, \leq$) for the boundaries of P To insert P into an IBS-tree with root R, the insertPredicate procedure is called which inserts the left end of the interval by calling addLeft(P, R) and the right end by calling addRight(P, R)

Below we will define addLeft and addRight These functions require the ability to determine whether everything in the right (or left) subtree of R will lie within P To help perform this test, we use functions rightUp(R) and leftUp(R) These functions are defined as follows

- rightUp(R) the lowest ancestor of R in the tree that contains R in its left subtree
- leftUp(R) the lowest ancestor of R in the tree that contains R in its right subtree

To find leftUp or rightUp for a node R, traverse upward from R and record leftUp or rightUp as necessary The procedure addLeft is shown in Figure 3 It recursively descends the tree, placing marks in the <, = and > slots of nodes as appropriate, and inserting a node in the tree if no node with the value of the interval's left boundary yet exists The procedure first tests to see if R is null, and if so makes R point to a new tree with one node containing a value equal to the left boundary of the interval Next, it checks for one of three possible cases

Case 1: If the value in R equals the left boundary of the interval, check to see if everything in the right subtree of R will lie within the interval If so, insert the identifier of the interval into the > slot of R Then, if the left boundary of the interval is defined using $\leq$, put the interval identifier in the = slot of R

Case 2: If the value of R is less than the interval's left boundary, call addLeft on the right subtree of R

Case 3: If the value of R is greater than the interval's left boundary, if the value lies in the interval, add the interval identifier to the = slot of R Next, if everything in the right subtree of R will lie in the interval, add the interval identifier to the > slot of R Finally, call addLeft on the left subtree of R

The procedure addRight(P, R) is symmetric to addLeft(P, R), so discussion of addRight is omitted

The above shows how to insert intervals into the tree In order to search the tree to find all intervals in the tree rooted at R that overlap a point X, and return a set of those predicates in S, the algorithm findIntervals(X,R,S) shown in Figure 4 is used The function findIntervals is called initially with arguments X, R and an initially empty set S If R is null, then no more matches are found and

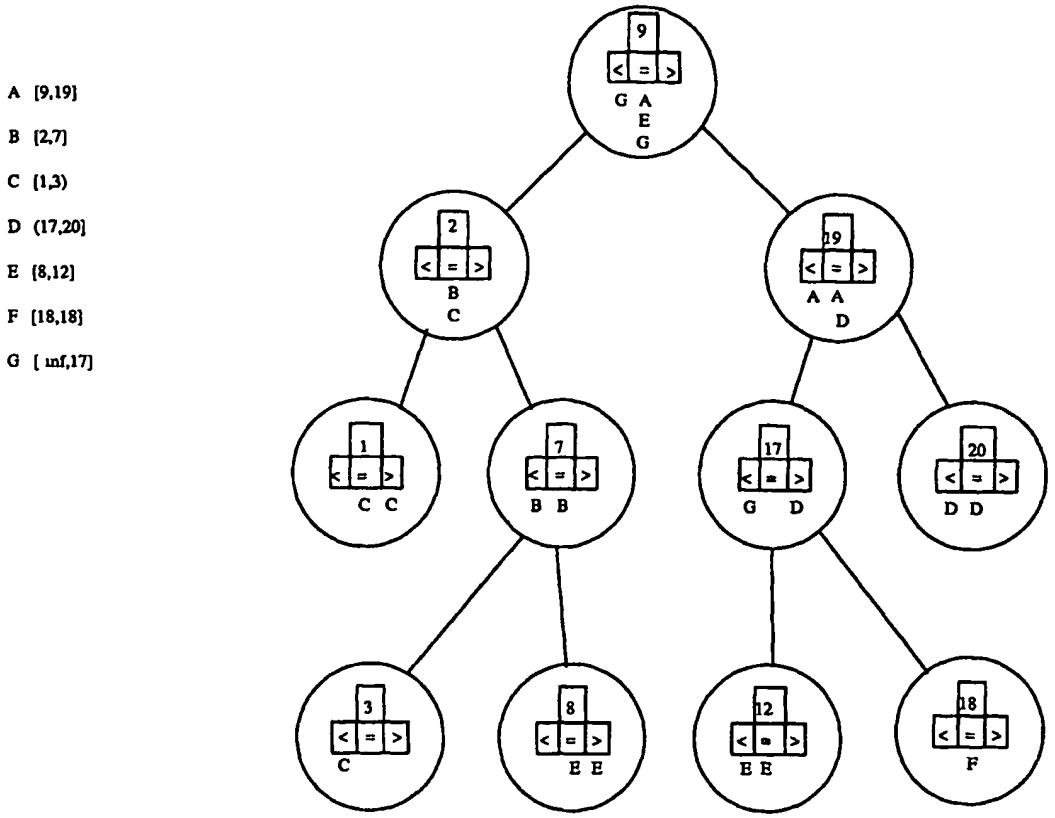


Figure 2 Example interval binary search tree for intervals shown

findIntervals returns If X is equal to the Value entry of node R, the elements of the $=$ set for R are added to the match set S If X is less than the Value of node R, the $<$ set of R is added to S, and the left subtree of R is searched If X is greater than the Value of node R, the $>$ set of R is added to S and the right subtree or R is searched

Deletion of nodes in an IBS-tree occurs only when an interval is deleted To delete an interval I, first delete all markers for I from the tree, using the reverse of the procedure for insertion Then delete the left and right endpoints of I if no other intervals have the same endpoint If an endpoint x is to be deleted, the following procedure is used

If x has no null child,

> Locate the predecessor of x by following right pointers from left(x) until finding a node with a null right child Call this predecessor node y Delete all markers set for intervals with end point y from the tree Store these intervals in temporary set T Swap the *values* of x and y, leaving the markers in their former locations

Make the pointer from the parent of x that points to x now point to either the non-null child of x, if there is one, or else to null Discard x Reinstall the markers for intervals in the set T

A justification of the correctness of this deletion procedure is given in [HC89] This concludes the discussion of basic operations on IBS-trees We now turn to methods for keeping IBS-trees balanced

4.3 Balancing Interval Binary Search Trees

The cost of the findIntervals algorithm depends on the height of the IBS-tree The algorithm for IBS-trees described above does not guarantee that the tree will be balanced Several balanced binary tree schemes have been proposed, including AVL trees [AL62], balanced binary trees (or red-black trees) [Bay72, GS78] and self-adjusting binary trees [Tar83] A common theme in these algorithms is the use of *rotations* to rebalance the tree In particular, during rebalancing operations, the balanced binary tree mechanisms cited make use of the single and double rotations shown in Figure 5 In the figure, lower case letters represent internal nodes, and upper case letters represent subtrees

There are symmetric variants of both single and double rotations which are not shown A double rotation is merely two applications of a single rotation Hence, to balance IBS-trees, all we need is a method for adjusting the marks on an

```
addLeft(P, R)
    do
        if R = Null
            set R to point to a new node with Value = P C_l fi
        if R Value = P C_l
            if everything in the right subtree of R will
                lie within P (i e rightUp(R) ≤ P C_r)
                    add the identifier of P to the '>' slot of R fi
            if P ρ_l is '≤'
                add the identifier of P to the '=' slot of R fi
        else if R Value < P C_l
            addLeft(P, R right)
        else if R Value > P C_l
            if R Value < P C_r (i e R Value is between P C_l and P C_r)
                add the identifier of P to the '=' slot of R fi
            if everything in the right subtree of R will
                lie within P (i e rightUp(R) ≤ P C_r)
                    add the identifier of P to the '>' slot of R fi
            addLeft(P, R left)
        fi
    od
```

Figure 3 Procedure to add the left end of an interval to an IBS-tree

```
findIntervals(X,R,S)
    do
        if R = Null        return
        else if X = R Value
            add the '=' set of R to S
            return
        else if X < R Value
            add the '<' set of R to S
            findIntervals(X,R left,S)
        else (X is > R Value)
            add the '>' set of R to S
            findIntervals(X,R right,S)
        fi
    od
```

Figure 4 Procedure for finding intervals that overlap a point X

IBS-tree during a single rotation so that the resulting tree is also a correct IBS-tree

Consider the single rotate-right operation shown in Figure 5 (a) and (b) The subtrees C and D, and the subtree rooted at x are unaffected during the operation, so no adjustment to them is required However, nodes y and z are modified to have different subtrees, so we must consider how to adjust the marks in the $<$, $=$ and $>$ fields of both after the rotation to leave the IBS-tree in a correct state

The following modifications to the marks are required during a rotation

1 Copy every mark from the $<$ slot of z to the $<$ and $=$ slots of y (this is necessary since having a mark for a predicate P in the $<$ slot of z implies that P matches every value in the left subtree of y, as well as y itself)

2 If a mark is in the $>$ slot of y but not in the $>$ slot of z before the rotation, then move the mark to the $<$ slot of z after the rotation This is necessary because values in the subtree C are covered by marks in the $>$ slot of y before the rotation, and must be covered by marks in the $<$ slot of z afterwards if they cannot be covered by a mark on y

3 If a mark is in the $>$ slot of y and the $>$ slot of z before the rotation, then remove the mark from the $=$ slot and $>$ slot of z after the rotation (this is necessary to avoid redundant locks on the values in subtree D)

Operations for each mark slot ($<, =, >$) on affected nodes y and z are summarized in Figure 6

In this section we have demonstrated that we can perform rotations about tree nodes and manipulate marks to restore the tree to be a correct IBS-tree The next section analyzes the performance of a balanced IBS-tree scheme that makes use of rotations

5 Performance Analysis

5.1 Analytical Performance Results

Assume that the AVL-tree scheme is used to maintain the balance of an IBS-tree [AL62] Each interval place $O(\log N)$ markers in the tree, for a worst-case storage requirement of $O(N \log N)$ Searching the tree to find all intervals that overlap a point X requires time $O(\log(N)+L)$ where N is the number of intervals indexed in the tree, and L is the number of intervals that overlap X This follows since traversing a path from root to leaf in the tree requires $O(\log N)$ time using a balanced tree scheme, and we must spend $O(1)$ time examining each of the L intervals retrieved

The cost of insertion and deletion of intervals in the tree is somewhat more difficult to calculate An important factor in

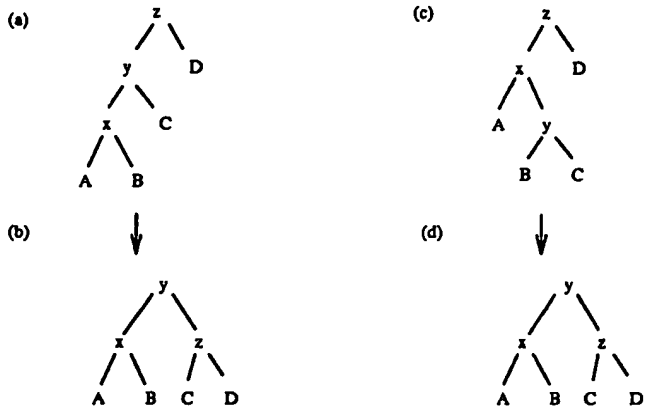


Figure 5 Single rotation (a,b) and double rotation (c,d)

	y	z
$<$	copy marks from $<$ of z	get marks from $>$ of y if necessary (see $> y$)
$=$	copy marks from $<$ of z	delete marks that are in both $>$ of y and $>$ of z
$>$	move marks to $<$ of z unless they appear in $>$ of z	delete marks that are in both $>$ of y and $>$ of z

Figure 6 Modifications to marks of each slot on affected nodes y and z required during a single rotation

the cost of insertions and deletions is the cost of doing a rotation Each insertion requires $O(1)$ rotations to rebalance the tree, and a deletion requires $O(\log N)$ rotations In [HC89] it is shown that the average cost of a rotation in an IBS tree is $O(\log N)$ Inserting an interval into an IBS tree requires $O(1)$ rotations for time $O(\log N)$, plus $O(\log N)$ insertions of markers into mark sets, each of which costs $O(\log N)$ if mark sets are maintained using auxiliary binary search trees, for a total cost of $O(\log^2 N)$ per insertion Deletion of an interval from an IBS tree also requires time $O(\log^2 N)$ for removing the markers from the tree, and time $O(log^2 N)$ for doing $O(\log N)$ rotations at a cost of $O(\log N)$ each, for a total time of $O(\log^2 N)$

The above discussion is an average case analysis of the cost of updating an IBS-tree with no restrictions on the width of intervals, or the extent to which intervals overlap An intriguing phenomenon is that when intervals in the tree do not overlap, only $O(N)$ markers are placed in the tree for a storage requirement of $O(N)$, and significantly reduced update cost Derivation of this result is left to the interested reader Since in many practical applications intervals have limited overlap, this gives hope that the actual time and space requirements for IBS-trees will be somewhat lower in practice than indicated by the analysis in this section

5.2 Empirical Performance Results

To get empirical figures on the performance of IBS-trees, the algorithm was implemented in C++ on a Sun SPARCstation 1 computer The balancing scheme using rotations was not implemented, but as with ordinary binary search trees, the tree is normally balanced if data is inserted in random order A series of IBS trees were created which contained N predicates for N between 0 and 1,000 A fraction a of predicates were simple points of the form *attribute = constant*, and the remaining fraction $1 - a$ were closed intervals The points and interval boundaries were drawn randomly from a uniform distribution of integers between 1 and 10,000 The length of the intervals was drawn randomly from a uniform distribution of integers between 1 and 1,000 The average times to insert a predicate for values of a=0, 5 and 1, and increasing values of N are shown in Figure 7 The average insertion cost was measured as the time to insert N predicates in an initially empty index, divided by N Since the test does not reflect any balancing cost, insertion times for balanced IBS-trees will be higher than shown in Figure 7 The average search time to find all predicates that match a value is plotted in Figure 8 for a=0, 5 and 1, and increasing values of N

As a basis of comparison for the IBS-tree algorithm, the cost of finding the predicates that match a value by traversing a linked list of predicates and testing each one against the value is shown in Figure 9 The cost curve for sequential search is always higher than for the IBS-tree, showing that the IBS-tree has quite low overhead

As expected, the insertion and search time curves for the IBS-tree both show logarithmic increase in search time as the number of intervals increases The difference between

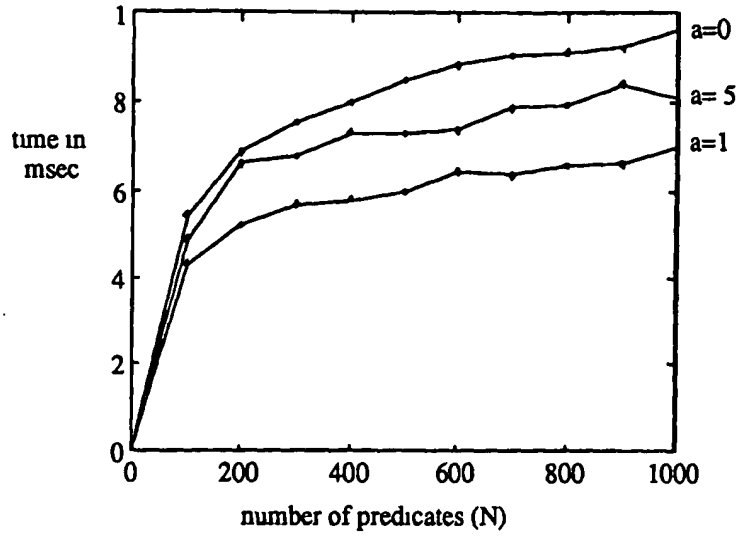


Figure 7 Average IBS-tree insertion times for α=0, 5 and 1

the curves for the different values of α (0, 5 and 1) are small, particularly for search time

When the IBS-tree is integrated into the overall predicate indexing scheme shown in Figure 1, predicate matching performance will depend on several factors, including

- the fraction of predicates that are non-indexable,
- the number of attributes per relation,
- the fraction of attributes that have one or more predicate clauses,
- the number of indexable predicate clauses per attribute

However, we can get an estimate for the time required to find matching predicates using the following assumptions

- hash search cost = 1 msec,
- fraction of predicates that are indexable = 90%,
- cost to test a predicate against a point in sequential search = 02 msec,
- average number of attributes per relation = 15,
- fraction of attributes per relation with 1 or more predicate clauses = 1/3,
- number of predicates per relation (N) = 200 (assuming that there are 200/5 = 40 predicates per attribute, the search cost in IBS-tree for one attribute is approximately 13 msec),
- cost to test an entire predicate against a tuple when a partial match is found = 05 msec,
- number of clauses per predicate = 2,
- average selectivity of each predicate clause = 1

The CPU usage times for operations shown above are reasonably close to the actual times for a Sun SPARCstation 1 In this scenario, the cost to search to find the partially matching predicates is the following

cost = hash cost
+ number of attributes searched
IBS-tree search cost
+ non-indexable predicate test cost

This yields the following numeric expression for cost

$$\text{cost} = 1 + 15\tfrac{1}{3} \quad 13 + (1 - 9) \quad 02 \quad 200$$
$$= 1 + 5 \quad 13 + 4 = 1\ 1 \text{ msec}$$

Since there are 200 predicates per relation, and the selectivity of the predicate clauses is 1, that means that 1 200 = 20 predicates must be tested after the initial search The time to test these is 05 20 = 1 msec Thus, the total time for predicate testing is 1 1 + 1 = 2 1 msec This is a fairly realistic number for the cost of finding all predicates that match a tuple using the algorithm presented in this paper with a moderate to large number of rules on a machine the speed of a SPARCstation 1 Given that this is a per-tuple CPU cost, the time is substantial, but should not be prohibitive Of course, these are CPU-only costs, and any increase in CPU speed will cause the predicate testing time to scale down accordingly

6 Conclusion

In this paper we have introduced a discrimination network structure for finding all members of a set of single-relation selection predicates that match a tuple It was argued that the structure will be small enough to fit in main memory for three reasons First, rules are a form of database schema, not data, and the size of the schema is normally relatively small Second, the largest rule-based expert systems built contain on the order of 10,000 rules, which is small enough to fit in main memory One would expect that the number of rules in a large database rules system application would be of comparable size Third, most systems applications that appear to require a very large number of rules can be redesigned to use a small number of rules plus additional

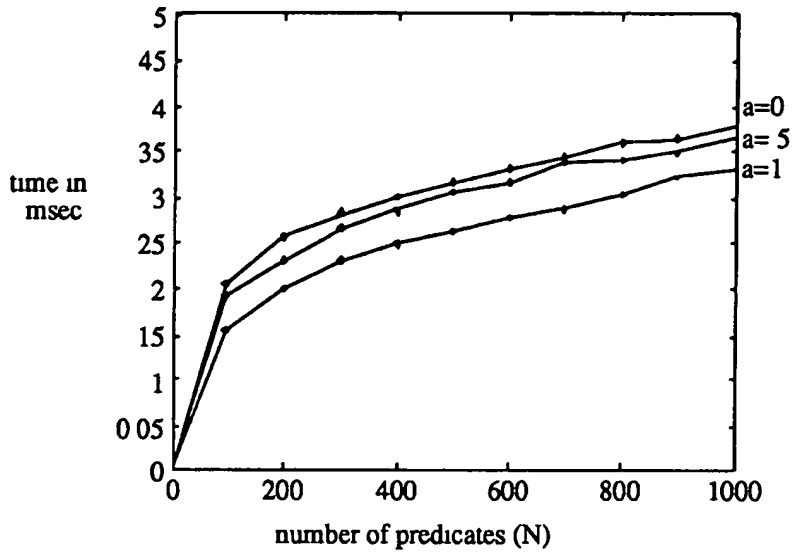


Figure 8 Average IBS-tree search times for a=0, 5 and 1

tuples or fields in the database Since the number of rules is expected to be small enough to fit in memory, a main-memory data structure was designed to take advantage of this

The key component of the algorithm proposed is the interval binary search tree, an extended binary search tree for indexing both interval and point data The IBS tree is related to the segment tree and interval tree, but in addition allows dynamic insertion and deletion of intervals and points while remaining balanced Analytical and empirical results show that the insertion and search performance of the IBS tree is almost as good as for an ordinary binary search tree The IBS tree or variations of it may be useful for other applications besides testing predicates, including VLSI CAD tools, geographic information systems, and other applications that deal with geometric data The IBS tree is useful anywhere an index for intervals is required which must be dynamically updatable

Although the intent of this paper was not to investigate parallelism, the algorithm proposed can easily be made to run significantly faster on a course-grain parallel machine such as a shared-memory multi-processor Parallelism can be achieved by searching the second-level index on each attribute of a tuple simultaneously, devoting a processor per attribute In addition, when brute force search is required, as in the case of non-indexable predicates and when doing the final predicate test, the set of predicates to be checked can be divided evenly among the available processors This could improve the performance of the algorithm by a factor nearly equal to the number of attributes searched in parallel (the initial hash cost is a per-tuple cost, and does not scale)

Considering topics for further research, an interesting area to investigate would be to implement several different techniques for dynamically indexing intervals, including 1-dimensional R-trees, IBS-trees, and priority search trees, and then compare their implementation complexity and time and space requirements Also, in the future we plan to work on developing an efficient structure to handle the join portion of rule predicates The discrimination network described in this paper will be used as the first layer of a two-layer network which will test both the selection and the join conditions of rules This two-layer approach is being implemented in the rule processing engine of the Ariel database system

References

[AL62] G M Adel'son-Vel'skii and E M Landis An algorithm for the organization of information *Soviet Math Dokl*, 3, 1962

[Bay72] R Bayer Symetric binary B-trees data structure and maintenance algorithms *Acta Informatica*, 1, 1972

[BO89] Virginia E Barker and Dennis E O'Connor Expert systems for configuration at Digital XCON and beyond *CACM*, 32(3), March 1989

[Col89] Larry Collins Informal survey of relational database applications at Wright-Patterson AFB 1989 (personal communication)

[DBB*88] U Dayal, B Blaustein, A Buchmann, et al The HiPAC project Combining active databases and timing constraints *SIGMOD Record*, 17(1) 51–70, March 1988

[DE88] Lois M L Delcambre and James N Etheredge The relational production language A production language for relational databases In *Proceedings of the Second International Conference on Expert Database Systems*, pages 153–162, April 1988

[For81] Charles L Forgy *OPS5 User's Manual* Technical Report CMU-CS-81-135, Carnegie-Mellon University, Pittsburgh, PA 15213, July 1981

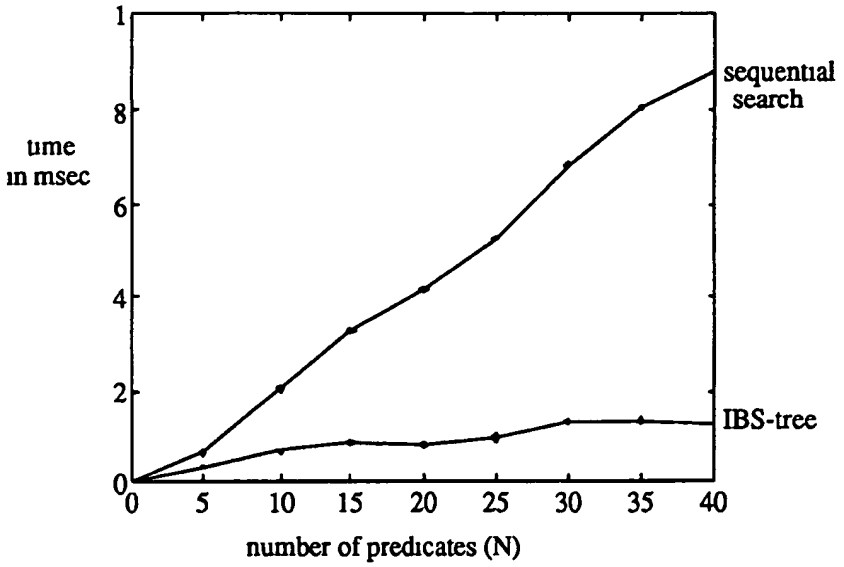


Figure 9 Predicate test cost for IBS-tree and sequential search

[For82] C L Forgy Rete A fast algorithm for the many pattern/many object pattern match problem *Artificial Intelligence*, 19 17–37, 1982

[GS78] L J Guibas and R Sedgewick A dichromatic framework for balanced binary trees In *Proc 19th Annual IEEE Symposium on Foundations of Computer Science*, 1978

[Gut84] A Guttman R-trees A dynamic index structure for spatial searching In *Proceedings of the 1984 ACM-SIGMOD Conference on Management of Data*, June 1984

[Han89] Eric N Hanson An initial report on the design of Ariel a DBMS with an integrated production rule system *SIGMOD Record*, 18(3), September 1989

[HC89] Eric N Hanson and Moez Chaabouni *The IBS Tree A Data Structure for Finding All Intervals That Overlap a Point* Technical Report, Wright State University, March 1989

[KS89] Michael A Kelly and Rudolph E Seviora An evaluation of DRete on CUPID for OPS5 In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1989

[McC85] Edward M McCreight Priority search trees *SIAM Journal of Computing*, 14(2) 257–278, 1985

[Mir87] Daniel P Miranker TREAT a better match algorithm for AI production systems In *Proceedings of AAAI 87 Conference on Artificial Intelligence*, pages 42–47, August 1987

[S*79] P Selinger et al Access path selection in a relational database management system In *Proceedings of the 1979 ACM-SIGMOD International Conference on Management of Data*, June 1979

[Sam88] Hanan Samet Hierarchical representations of collections of small rectangles *ACM Computing Surveys*, 20(4) 271–309, December 1988

[Sam90] Hanan Samet *The Design and Analysis of Spatial Data Structures* Addision Wesley, 1990

[SHP88] Michael Stonebraker, Eric Hanson, and Spiros Potamianos The POSTGRES rule manager *IEEE Transactions on Software Engineering*, 14(7) 897–907, July 1988

[SLR89] Timos Sellis, Chih-Chen Lin, and Louiqa Raschid Data intesive production systems the DIPS approach *SIGMOD Record*, September 1989

[SSH86] M Stonebraker, T Sellis, and E Hanson An analysis of rule indexing implementations in data base systems In *Proceedings of the First Annual Conference on Expert Database Systems*, April 1986

[Tar83] Robert Endre Tarjan *Data Structures and Network Algorithms* Society for Industrial and Applied Mathematics, 1983

ACM PORTAL DIGITAL LIBRARY

> home > about > feedback > login

US Patent & Trademark Office

Try the *new* Portal design

Give us your opinion after using it.

Citation

International Conference on Management of Data and Symposium on Principles of Database Systems >archive

Proceedings of the 1990 ACM SIGMOD international conference on Management of data >toc

1990 , Atlantic City, New Jersey, United States

A predicate matching algorithm for database rule systems

> Also published in ...

Authors

Eric N. Hanson
Moez Chaabouni
Chang-Ho Kim
Yu-Wang Wang

Sponsor

SIGMOD : ACM Special Interest Group on Management of Data

Publisher

ACM Press New York, NY, USA

Pages: 271 - 280 Series-Proceeding-Article
Year of Publication: 1990
ISSN:0163-5808

doi> http://doi.acm.org/10.1145/93597.98736 (Use this link to Bookmark this page)

> full text > abstract > references > citings > index terms > peer to peer

> Discuss > Similar > Review this Article Save to Binder

> BibTex Format

↑ **FULL TEXT:** Access Rules

pdf 1.08 MB

↑ ABSTRACT

Forward-chaining rule systems must test each newly asserted fact against a collection of predicates to find those rules that match the fact. Expert system rule engines use a simple combination of hashing and sequential search for this matching. We introduce an algorithm for finding the

matching predicates that is more efficient than the standard algorithm when the number of predicates is large. We focus on equality and inequality predicates on totally ordered domains. This algorithm is well-suited for database rule systems, where predicate-testing speed is critical. A key component of the algorithm is the interval binary search tree (IBS-tree). The IBS-tree is designed to allow efficient retrieval of all intervals (e.g. range predicates) that overlap a point, while allowing dynamic insertion and deletion of intervals. The algorithm could also be used to improve the performance of forward-chaining inference engines for large expert systems applications.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

AL62 G M Adel'son-Vel'skn and E M Landis An algorithm for the organization of reformation Somet Math Dokl, 3, 1962

Bay72 P~ Bayer Symetnc binary B-trees data structure and maintenance algorithms Acta InJormat- 8ca, 1, 1972

BO89 Virginia E. Barker , Dennis E. O'Connor , Judith Bachant , Elliot Soloway, Expert systems for configuration at Digital: XCON and beyond, Communications of the ACM, v.32 n.3, p.298-318, March 1989

Col89 Larry Colhns Informal survey of relational database apphcatlons at Wright-Patterson AFB 1989 (personal commumcatlon)

DBB*88 M. J. Carey , M. Livny , R. Jauhari, The HiPAC project: combining active databases and timing constraints, ACM SIGMOD Record, v.17 n.1, p.51-70, March, 1988

DE88 Lois M L Delcambre and James N Etheredge The relational production language A production language for relational databases In Proceedmgs of the Second International Conference on Expert Database Systems, pages 153-162, April 1988

For81 Charles L Forgy OPS5 User's Manual Techmcal Report CMU-CS-81-135, Carnegm-Mellon Umversity, Pittsburgh, PA 15213, 3uly 1981

For82 C L Forgy Rete A fast algorithm for the many pattern/many object pattern match problem Art2ficsal Intellsgence, 19 17-37, 1982

GS78 L J Gulbas and R Sedgewlck A dichromatic framework for balanced binary trees In Proc 19th Annual IEEE Sympos,um on Foundat, ons of Computer Sczence, 1978

Gut84 A Guttman R-trees A dynamic mdexstructure for spahal searching In Proceed, ngs of the 1984 A CM-SIGMOD Conference on Management of Data, June 1984

Han89 E. N. Hanson, An initial report on the design of Ariel DBMS with an integrated production rule system, ACM SIGMOD Record, v.18 n.3, p.12-19, Sep. 1989

HC89 Enc N Hanson and Moez Chaaboum The IBS Tree A Data Structure .for Fsn&ng All Intervals That Overlap a Posnt Technical Report, Wright State Umversity, March 1989

KS89 Michael A Kelly and Rudolph E Sevlora An evaluahon of DRete on CUPID for OPS5 In Procee&ngs o/ the Eleventh Internatzonal Josnt Conference on Artsficsal Intellzgence, 1989

McC85 Edward M McCrelght Pnonty search trees SIAM Journal o/ Computzng, 14(2) 257-278, 1985

Mir87 Darnel P Mlranker TREAT a better match algonthm for AI produchon systems In Proceed-,ngs of AAAI 87 Conference on Artsfic,al intelligence, pages 42-47, August 1987

S*79 P Sehnger et al Access path selection in a relational database management system In Proceedzngs of the 1979 A CM-SIGMOD Internat,onal Conference on Management of Data, June 1979

Sam88 Hanan Samet, Hierarchical representations of collections of small rectangles, ACM Computing Surveys (CSUR), v.20 n.4, p.271-309, Dec. 1988

Sam90 Hanan Samet, The design and analysis of spatial data structures, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1990

SHP88 Michael Stonebraker , Eric N. Hanson , Spyros Potamianos, The POSTGRES Rule Manager, IEEE Transactions on Software Engineering, v.14 n.7, p.897-907, July 1988

SLR89 T. Sellis , C. Lin , L. Raschid, Data intensive production systems: the DIPS approach, ACM SIGMOD Record, v.18 n.3, p.52-58, Sep. 1989

SSH86 M Stonebraker, T Selhs, and E Hanson An analysis of rule mderang Implementatmns in data base systems In Proceed,ngs o/ the F, rst Annual Conference on Expert Database Systems, April 1986

Tar83 Robert Endre Tarjan, Data structures and network algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983

↑ CITINGS *11*

Hubert Ka Yau Leung , Hans-Arno Jacobsen, Efficient matching for state-persistent publish/subscribe systems, Proceedings of the 2003 conference of the Centre for Advanced Studies conference on Collaborative research, p.182-196, October 06-09, 2003, Toronto, Ontario, Canada

Moez Chaabouni , Soon M. Chung, Discrimination network for rule condition matching in object-oriented database rule systems, Proceedings of the 1995 ACM symposium on Applied computing, p.51-55, February 26-28, 1995, Nashville, Tennessee, United States

Marcos K. Aguilera , Robert E. Strom , Daniel C. Sturman , Mark Astley , Tushar D. Chandra, Matching events in a content-based subscription system, Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing, p.53-61, May 04-06, 1999, Atlanta, Georgia, United States

Moez Chaabouni , Soon M. Chung, The point-range tree: a data structure for indexing intervals, Proceedings of the 1993 ACM conference on Computer science, p.453-460, February 16-18, 1993, Indianapolis, Indiana, United States

Kun-Lung Wu , Philip S. Yu, Efficient query monitoring using adaptive multiple key hashing, Proceedings of the eleventh international conference on Information and knowledge management, November 04-09, 2002, McLean, Virginia, USA

Françoise Fabret , H. Arno Jacobsen , François Llirbat , João Pereira , Kenneth A. Ross , Dennis Shasha, Filtering algorithms and implementation for very fast publish/subscribe systems, ACM

SIGMOD Record, v.30 n.2, p.115-126, June 2001

Ouri Wolfson , Hasanat M. Dewan , Salvatore J. Stolfo , Yechiam Yemini, Incremental evaluation of rules and its relationship to parallelism, ACM SIGMOD Record, v.20 n.2, p.78-87, June 1991

Walid Rjaibi , Klaus R. Dittrich , Dieter Jaepel, Event matching in symmetric subscription systems, Proceedings of the 2002 conference of the Centre for Advanced Studies on Collaborative research, p.9, September 30-October 03, 2002, Toronto, Ontario, Canada

Eric N. Hanson, Rule condition testing and action execution in Ariel, ACM SIGMOD Record, v.21 n.2, p.49-58, June 1, 1992

Eric N. Hanson , Tina M. Harvey , Mark A. Roth, Experiences in DBMS implementation using an object-oriented persistent programming language and a database toolkit, ACM SIGPLAN Notices, v.26 n.11, p.314-328, Nov. 1991

Lars Baekgaard , Leo Mark, Incremental computation of nested relational query expressions, ACM Transactions on Database Systems (TODS), v.20 n.2, p.111-148, June 1995

↑ INDEX TERMS

Primary Classification:

I. Computing Methodologies
- ↳ **I.2** ARTIFICIAL INTELLIGENCE
  - ↳ **I.2.4** Knowledge Representation Formalisms and Methods
    - ↳ **Subjects:** Representations (procedural and rule-based)

Additional Classification:

F. Theory of Computation
- ↳ **F.2** ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY
  - ↳ **F.2.2** Nonnumerical Algorithms and Problems
    - ↳ **Subjects:** Sorting and searching

H. Information Systems
- ↳ **H.2** DATABASE MANAGEMENT

I. Computing Methodologies
- ↳ **I.2** ARTIFICIAL INTELLIGENCE
  - ↳ **I.2.8** Problem Solving, Control Methods, and Search
    - ↳ **Subjects:** Graph and tree search strategies; Heuristic methods

General Terms:
Algorithms, Performance

↑ Peer to Peer - Readers of this Article have also read:

- ◆ Data structures for quadtree approximation and compression
  Communications of the ACM 28, 9

Hanan Samet

- A hierarchical single-key-lock access control using the Chinese remainder theorem
  Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing
  Kim S. Lee , Huizhu Lu , D. D. Fisher

- 3D representations for software visualization
  Proceedings of the 2003 ACM symposium on Software visualization
  Andrian Marcus , Louis Feng , Jonathan I. Maletic

- Probabilistic surfaces: point based primitives to show surface uncertainty
  Proceedings of the conference on Visualization '02
  Gevorg Grigoryan , Penny Rheingans

- Efficient simplification of point-sampled surfaces
  Proceedings of the conference on Visualization '02
  Mark Pauly , Markus Gross , Leif P. Kobbelt

↑ **This Article has also been published in:**

- **ACM SIGMOD Record**
  Volume 19 , Issue 2 (June 1990)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.

ACM PORTAL DIGITAL LIBRARY

> home > about > feedback > login

US Patent & Trademark Office

Try the *new* Portal design

Give us your opinion after using it.

Citation

International Conference on Management of Data and Symposium on Principles of Database Systems >archive

Proceedings of the 1990 ACM SIGMOD international conference on Management of data >toc

1990 , Atlantic City, New Jersey, United States

A predicate matching algorithm for database rule systems

> Also published in ...

Authors
Eric N. Hanson
Moez Chaabouni
Chang-Ho Kim
Yu-Wang Wang

Sponsor
SIGMOD : ACM Special Interest Group on Management of Data

Publisher
ACM Press New York, NY, USA

Pages: 271 - 280 Series-Proceeding-Article
Year of Publication: 1990
ISSN:0163-5808

doi> http://doi.acm.org/10.1145/93597.98736 (Use this link to Bookmark this page)

> full text > abstract > references > citings > index terms > peer to peer

> Discuss > Similar > Review this Article Save to Binder

> BibTex Format

↑ **FULL TEXT:** Access Rules

pdf 1.08 MB

↑ **ABSTRACT**

Forward-chaining rule systems must test each newly asserted fact against a collection of predicates to find those rules that match the fact. Expert system rule engines use a simple combination of hashing and sequential search for this matching. We introduce an algorithm for finding the

matching predicates that is more efficient than the standard algorithm when the number of predicates is large. We focus on equality and inequality predicates on totally ordered domains. This algorithm is well-suited for database rule systems, where predicate-testing speed is critical. A key component of the algorithm is the interval binary search tree (IBS-tree). The IBS-tree is designed to allow efficient retrieval of all intervals (e.g. range predicates) that overlap a point, while allowing dynamic insertion and deletion of intervals. The algorithm could also be used to improve the performance of forward-chaining inference engines for large expert systems applications.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

AL62 G M Adel'son-Vel'skn and E M Landis An algorithm for the organization of reformation Somet Math Dokl, 3, 1962

Bay72 P~ Bayer Symetnc binary B-trees data structure and maintenance algorithms Acta InJormat- 8ca, 1, 1972

BO89 Virginia E. Barker , Dennis E. O'Connor , Judith Bachant , Elliot Soloway, Expert systems for configuration at Digital: XCON and beyond, Communications of the ACM, v.32 n.3, p.298-318, March 1989

Col89 Larry Colhns Informal survey of relational database apphcatlons at Wright-Patterson AFB 1989 (personal commumcatlon)

DBB*88 M. J. Carey , M. Livny , R. Jauhari, The HiPAC project: combining active databases and timing constraints, ACM SIGMOD Record, v.17 n.1, p.51-70, March, 1988

DE88 Lois M L Delcambre and James N Etheredge The relational production language A production language for relational databases In Proceedmgs of the Second International Conference on Expert Database Systems, pages 153-162, April 1988

For81 Charles L Forgy OPS5 User's Manual Techmcal Report CMU-CS-81-135, Carnegm-Mellon Umversity, Pittsburgh, PA 15213, 3uly 1981

For82 C L Forgy Rete A fast algorithm for the many pattern/many object pattern match problem Art2ficsal Intellsgence, 19 17-37, 1982

GS78 L J Gulbas and R Sedgewlck A dichromatic framework for balanced binary trees In Proc 19th Annual IEEE Sympos,um on Foundat, ons of Computer Sczence, 1978

Gut84 A Guttman R-trees A dynamic mdexstructure for spahal searching In Proceed, ngs of the 1984 A CM-SIGMOD Conference on Management of Data, June 1984

Han89 E. N. Hanson, An initial report on the design of Ariel DBMS with an integrated production rule system, ACM SIGMOD Record, v.18 n.3, p.12-19, Sep. 1989

HC89 Enc N Hanson and Moez Chaaboum The IBS Tree A Data Structure .for Fsn&ng All Intervals That Overlap a Posnt Technical Report, Wright State Umversity, March 1989

KS89 Michael A Kelly and Rudolph E Sevlora An evaluahon of DRete on CUPID for OPS5 In Procee&ngs o/ the Eleventh Internatzonal Josnt Conference on Artsficsal Intellzgence, 1989

McC85 Edward M McCrelght Pnonty search trees SIAM Journal o/ Computzng, 14(2) 257-278, 1985

Mir87 Darnel P Mlranker TREAT a better match algonthm for AI produchon systems In Proceed-,ngs of AAAI 87 Conference on Artsfic,al intelligence, pages 42-47, August 1987

S*79 P Sehnger et al Access path selection in a relational database management system In Proceedzngs of the 1979 A CM-SIGMOD Internat,onal Conference on Management of Data, June 1979

Sam88 Hanan Samet, Hierarchical representations of collections of small rectangles, ACM Computing Surveys (CSUR), v.20 n.4, p.271-309, Dec. 1988

Sam90 Hanan Samet, The design and analysis of spatial data structures, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, 1990

SHP88 Michael Stonebraker , Eric N. Hanson , Spyros Potamianos, The POSTGRES Rule Manager, IEEE Transactions on Software Engineering, v.14 n.7, p.897-907, July 1988

SLR89 T. Sellis , C. Lin , L. Raschid, Data intensive production systems: the DIPS approach, ACM SIGMOD Record, v.18 n.3, p.52-58, Sep. 1989

SSH86 M Stonebraker, T Selhs, and E Hanson An analysis of rule mderang Implementatmns in data base systems In Proceed,ngs o/ the F, rst Annual Conference on Expert Database Systems, April 1986

Tar83 Robert Endre Tarjan, Data structures and network algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983

↑ CITINGS *11*

Hubert Ka Yau Leung , Hans-Arno Jacobsen, Efficient matching for state-persistent publish/subscribe systems, Proceedings of the 2003 conference of the Centre for Advanced Studies conference on Collaborative research, p.182-196, October 06-09, 2003, Toronto, Ontario, Canada

Moez Chaabouni , Soon M. Chung, Discrimination network for rule condition matching in object-oriented database rule systems, Proceedings of the 1995 ACM symposium on Applied computing, p.51-55, February 26-28, 1995, Nashville, Tennessee, United States

Marcos K. Aguilera , Robert E. Strom , Daniel C. Sturman , Mark Astley , Tushar D. Chandra, Matching events in a content-based subscription system, Proceedings of the eighteenth annual ACM symposium on Principles of distributed computing, p.53-61, May 04-06, 1999, Atlanta, Georgia, United States

Moez Chaabouni , Soon M. Chung, The point-range tree: a data structure for indexing intervals, Proceedings of the 1993 ACM conference on Computer science, p.453-460, February 16-18, 1993, Indianapolis, Indiana, United States

Kun-Lung Wu , Philip S. Yu, Efficient query monitoring using adaptive multiple key hashing, Proceedings of the eleventh international conference on Information and knowledge management, November 04-09, 2002, McLean, Virginia, USA

Françoise Fabret , H. Arno Jacobsen , François Llirbat , João Pereira , Kenneth A. Ross , Dennis Shasha, Filtering algorithms and implementation for very fast publish/subscribe systems, ACM

SIGMOD Record, v.30 n.2, p.115-126, June 2001

Ouri Wolfson , Hasanat M. Dewan , Salvatore J. Stolfo , Yechiam Yemini, Incremental evaluation of rules and its relationship to parallelism, ACM SIGMOD Record, v.20 n.2, p.78-87, June 1991

Walid Rjaibi , Klaus R. Dittrich , Dieter Jaepel, Event matching in symmetric subscription systems, Proceedings of the 2002 conference of the Centre for Advanced Studies on Collaborative research, p.9, September 30-October 03, 2002, Toronto, Ontario, Canada

Eric N. Hanson, Rule condition testing and action execution in Ariel, ACM SIGMOD Record, v.21 n.2, p.49-58, June 1, 1992

Eric N. Hanson , Tina M. Harvey , Mark A. Roth, Experiences in DBMS implementation using an object-oriented persistent programming language and a database toolkit, ACM SIGPLAN Notices, v.26 n.11, p.314-328, Nov. 1991

Lars Baekgaard , Leo Mark, Incremental computation of nested relational query expressions, ACM Transactions on Database Systems (TODS), v.20 n.2, p.111-148, June 1995

↑ INDEX TERMS

Primary Classification:

I. Computing Methodologies
- ↳ **I.2** ARTIFICIAL INTELLIGENCE
  - ↳ **I.2.4** Knowledge Representation Formalisms and Methods
    - ↳ **Subjects:** Representations (procedural and rule-based)

Additional Classification:

F. Theory of Computation
- ↳ **F.2** ANALYSIS OF ALGORITHMS AND PROBLEM COMPLEXITY
  - ↳ **F.2.2** Nonnumerical Algorithms and Problems
    - ↳ **Subjects:** Sorting and searching

H. Information Systems
- ↳ **H.2** DATABASE MANAGEMENT

I. Computing Methodologies
- ↳ **I.2** ARTIFICIAL INTELLIGENCE
  - ↳ **I.2.8** Problem Solving, Control Methods, and Search
    - ↳ **Subjects:** Graph and tree search strategies; Heuristic methods

General Terms:
Algorithms, Performance

↑ Peer to Peer - Readers of this Article have also read:

- ◆ Data structures for quadtree approximation and compression
  Communications of the ACM 28, 9

Hanan Samet

- A hierarchical single-key-lock access control using the Chinese remainder theorem
  Proceedings of the 1992 ACM/SIGAPP Symposium on Applied computing
  Kim S. Lee , Huizhu Lu , D. D. Fisher

- 3D representations for software visualization
  Proceedings of the 2003 ACM symposium on Software visualization
  Andrian Marcus , Louis Feng , Jonathan I. Maletic

- Probabilistic surfaces: point based primitives to show surface uncertainty
  Proceedings of the conference on Visualization '02
  Gevorg Grigoryan , Penny Rheingans

- Efficient simplification of point-sampled surfaces
  Proceedings of the conference on Visualization '02
  Mark Pauly , Markus Gross , Leif P. Kobbelt

↑ **This Article has also been published in:**

- **ACM SIGMOD Record**
  Volume 19 , Issue 2 (June 1990)

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.